National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Services des thèses canadiennes

Ottawa, Canada
K1A 0N4

CANADIAN THESES
THÈSES CANADIENNES

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30.

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

Canada

NL-339 (r.86/06)

THE UNIVERSITY OF ALBERTA

Characteristics of Multirate Regulators

by

(C)

Arlen D. Schmidt

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF Master of Science

Department of Electrical Engineering

EDMONTON, ALBERTA

Fall, 1986

Permission has been granted to the National Library of Canada to microfilm this thesis and to lend or sell copies of the film.

The author (copyright owner) has reserved other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without his/her written permission.

L'autorisation a été accordée à la Bibliothèque nationale du Canada de microfilmer cette thèse et de prêter ou de vendre des exemplaires du film.

L'auteur (titulaire du droit d'auteur) se réserve les autres droits de publication; ni la thèse ni de longs extraits de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation écrite.

ISBN 0-315-32307-8

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR Arlen D. Schmidt

TITLE OF THESIS Characteristics of Multirate Regulators

DEGREE FOR WHICH THESIS WAS PRESENTED Master of Science

YEAR THIS DEGREE GRANTED Fall, 1986

Permission is hereby granted to THE UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

(SIGNED) A. Schmidt.....

PERMANENT ADDRESS:

9716 - 88 Avenue

Edmonton, Alberta

T6E 2P9

DATED September 15 1986

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled Characteristics of Multirate Regulators submitted by Arlen D. Schmidt in partial fulfilment of the requirements for the degree of Master of Science.

.....

Supervisor

.....

.....

Date SEPTEMBER 15, 1986

ABSTRACT

The optimal multirate regulator suffers from a complex structure and the lack of a simple algorithm for its solution. These drawbacks are addressed by developing an improved algorithm for the solution of the optimal multirate regulator and by examining methods for simplifying the regulator structure.

First, the design of an optimal multirate regulator is reviewed and an improved algorithm for its solution is developed. The new algorithm is simpler than those published in recent papers as a result of its unique treatment of the additional states needed to hold control values. It is shown that the resulting optimal regulator is periodically time varying and uses the plant state and the held controls in the calculation of all new controls.

The question of regulator structure is addressed by proposing a variety of suboptimal multirate regulators having simpler structures than their optimal counterpart. Whether the more complex optimal regulator is needed in any given situation can be decided by examining the performance of the suboptimal regulators.

The design methods for both the optimal and suboptimal regulators are applied to two example systems. These examples quantify the difference in performance between the different regulator structures. By using this information, the system designer should be able to make an intelligent cost/performance tradeoff decision.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my supervisor and friend Dr. Raymond E. Rink. His continual guidance and deep insight have provided me with an educational experience that will guide me in both my professional and private life. Although there are many virtues I would like to possess, my greatest desire is to develop the patience and kindness I was repeatedly shown by Dr. Rink.

I would also like to thank the staff of Intera Technologies. The flexibility and patience of my supervisors and the interest and encouragement of my colleagues provided much needed inspiration.

Finally, many thanks to Dr. V.G. Gourishankar and Dr. R.K. Wood for their careful review of this thesis.

Table of Contents

Chapter		Page
1.	INTRODUCTION	1
	1.1 Justification	1
	1.2 Scope of Thesis	3
	1.3 New Results	4
	1.4 Thesis Outline	7
2.	THE MULTIRATE SYSTEM	9
	2.1 Introduction	9
	2.2 Frequency Domain Techniques	9
	2.3 Time Domain Techniques	10
	2.4 Problem Statement	11
	2.5 The Multirate Sampling Scheme	15
	2.6 Summary	16
3.	THE OPTIMAL MULTIRATE REGULATOR	18
	3.1 Introduction	18
	3.2 Problem Statement	18
	3.3 Method of Solution	19
	3.4 The Multistage System	23
	3.5 Conversion to Time Invariant Form	28
	3.6 The Solution of the Optimal Multirate Regulator	33
	3.7 Calculation of the Optimal Feedback Gains	35
	3.8 Structure of the Optimal Multirate Regulator	36
	3.9 Summary	40
4.	THE OPTIMAL CONSTRAINED MULTIRATE REGULATOR	42
	4.1 Introduction	42
	4.2 Method of Solution	43
	4.3 The Optimal Plant-State-Only Regulator	46

4.4 The Optimal Constant-Gain Regulator 47

4.5 The Optimal Constant-Gain Plant-State-Only Regulator 48

4.6 Discussion of Characteristics 48

4.7 Summary 49

5. SUBOPTIMAL MULTIRATE REGULATORS 51

5.1 Introduction 51

5.2 The Suboptimal Plant-State-Only Regulator 51

5.3 The Averaged-Gains Regulator 55

5.4 The Separated-Modes Regulator 56

5.5 Summary 57

6. MULTIRATE SYSTEM DESIGN AND ANALYSIS 59

6.1 Introduction 59

6.2 Conversion to Discrete-time Form 60

6.3 The Discrete-time Riccati Equation 69

6.4 Solution of the Algebraic Riccati Equation 71

6.5 Conversion to Diagonal Form 72

6.6 System Response to Process Noise 75

6.7 Calculation of the Cost Functional 77

6.8 Summary 79

7. EXAMPLE #1 - THE INVERTED PENDULUM 80

7.1 Introduction 80

7.2 The Inverted Pendulum 81

7.3 A Typical Inverted Pendulum Regulator 88

7.4 Length Effects 95

7.5 Mass Ratio Effects 99

7.6 Summary 104

8. EXAMPLE #2 - THE OSCILLATING CARTS 106

8.1 Introduction 106
8.2 The Oscillating Carts 107
8.3 Performance of the Multirate Regulators 110
8.4 Summary 113
9. CONCLUSION 115
9.1 Summary and Conclusions 115
9.2 Suggestions for Further Research 120
BIBLIOGRAPHY 125
APPENDIX A 128
APPENDIX B 176
APPENDIX C 182
APPENDIX D 184

List of Tables

Table		Page
7.1	Feedback Gains for a Typical Inverted Pendulum System	91
7.2	Multirate Regulator Performance for a Typical Inverted Pendulum System	93
7.3	Regulator Performance as a Function of Length	96
7.4	Regulator Performance as a Function of Mass Ratio	100
8.1	Regulator Performance for the Oscillating Cart System	111

List of Figures

Figure		Page
2.1	Multirate Regulator for System Having Two Control Inputs	13
3.1	State Propagation Over One Control Period	21
6.1	Discrete-time Control of a Continuous-time Plant	61
7.1	The Inverted Pendulum	82
7.2	Regulator Performance as a Function of Length	97
7.3	Regulator Performance as a Function of Mass Ratio	101
8.1	The Oscillating Carts	108

LIST OF SYMBOLS

SCALARS

α	steepest descent algorithm parameter (section 4.2) †
Δ	scalar variable (section 6.2)
j_{cp}	expected cost over one control period (section 3.4)
J	expected value of cost functional (section 2.4)
k	number of cycles per control period (section 2.5)
l	number of new controls in control period (section 3.5)
m	number of control inputs (section 2.4)
n	number of plant states (section 2.4)
r	number of disturbance inputs (section 2.4)
t_i	discrete time variable (section 2.4)
T	control period in seconds (section 2.4)
τ	continuous time variable (section 2.4)
$u_{i,j}$	(i,j) element of UP (section 3.5)

VECTORS

u	control vector (section 2.4)
u_f	fast control vector (section 2.4)
u_h	held control vector (section 2.4)
u_n	new control vector (section 2.4)
u_s	slow control vector (section 2.4)
U	multistage control vector (section 3.3)
U_m	reduced multirate control vector (section 3.5)
w	process noise vector (section 2.4)
w_a	augmented process noise vector (section 4.2)

---

†The section in which the symbol is first introduced is included in brackets.

x state vector (section 2.4)

$\bar{x}$ mean state vector (section 6.7)

x_a augmented state vector (section 3.8)

MATRICES

A real matrix variable (section 3.5)

A' real matrix variable (section 3.5)

C cycle feedback gain matrix (section 3.3)

C' cycle feedback gain matrix (section 4.3)

C'' cycle feedback gain matrix (section 4.4)

C''' cycle feedback gain matrix (section 4.5)

C expanded feedback gain matrix (section 5.2)

C_m optimal multirate feedback gain matrix (section 5.2)

E folding matrix (section 3.5)

E_a augmented folding matrix (section 3.5)

F continuous-time state transition matrix (section 2.4)

F_2 real matrix variable (section 6.2)

F_3 real matrix variable (section 6.2)

G continuous-time control distribution matrix (section 2.4)

G_v continuous-time disturbance distribution matrix (section 2.4)

G_2 real matrix variable (section 6.2)

Γ discrete-time control distribution matrix (section 2.4)

Γ_a augmented control distribution matrix (section 3.8)

Γ_h held control distribution matrix (section 3.8)

Γ_k control distribution matrix for one control period (section 3.3)

Γ_n new control distribution matrix (section 3.8)

H_c control Hamiltonian (section 6.4)

K general control period feedback gain matrix (section 3.3)

K Riccati equation solution (section 6.3)

$\hat{K}$ algebraic Riccati equation solution (section 6.4)

M discrete-time cross weighting matrix (section 2.4)

ϕ discrete-time state transition matrix (section 2.4)

ϕ_a augmented state transition matrix (section 3.8)

Φ state transition matrix for one control period (section 3.3)

Q discrete-time state weighting matrix (section 2.4)

$\hat{Q}_a$ augmented closed-loop state weighting matrix (section 4.2)

Q_c real matrix variable (section 6.2)

Q_A continuous-time state weighting matrix (section 2.4)

R discrete-time control weighting matrix (section 2.4)

R_A continuous-time control weighting matrix (section 2.4)

S state covariance (section 6.4)

S_a augmented state covariance (section 4.2)

T real matrix variable (section 3.4)

$\mathbf{T}$ real matrix variable (sections 3.4 and 3.6)

θ control extraction matrix (section 3.4)

θ_h hold state transition matrix for held controls (section 3.8)

θ_n hold state transition matrix for new controls (section 3.8)

U matrix of Schur vectors (section 6.4)

UP multirate control update matrix (section 2.5)

W continuous-time process noise covariance (section 6.2)

W_d discrete-time process noise covariance (section 3.2)

Ω real matrix variable (section 6.5)

X state covariance (section 6.7)

OTHERS

cg	constant-gain
pso	plant-state-only
cgpso	constant-gain plant-state-only
LQG	linear-quadratic-Gaussian
ZOH	zero-order hold
$N(\mu,\sigma^2)$	normal distribution; mean = μ, covariance = σ^2
cov(·)	covariance of
DIM(·)	dimension of
E{ · }	expected value of
tr{ · }	trace of

Chapter 1

INTRODUCTION

1.1 Justification

Much research has been devoted to the question of sample rate selection for digital control systems. It is usually true that the highest quality control is obtained by using the highest possible sample rate. Unfortunately, the sample rate is usually limited by the computational capabilities of the control computer and increasing the sample rate often requires the purchase of higher performance computational equipment. In order to choose equipment that provides an optimal cost/benefit ratio, the designer must examine the lower limits on the sample rate. These limits are determined by considerations such as: closed-loop bandwidth, time response to control inputs, roughness of the response to control inputs, sensitivity to noise, and sensitivity to plant parameter variations. With this information, a control system designer should be able to select the lowest sample rate necessary to meet all requirements and thus choose the most economical control computer.

Using the tools of modern control theory, it is possible to design controllers for complicated systems of high order. State space design techniques have unified the design of these systems and give the (deceptively) simple appearance of requiring no more than the application of cookbook formulae. This approach is rarely satisfactory and the designer is usually forced to examine the

characteristics of the plant in some detail in order to choose the best practical controller. These high order systems commonly accept many input signals and provide a variety of output signals. By considering each signal individually, the designer will often find a large difference in the sample rates necessary to meet system requirements. It is generally the case that the signals requiring higher sampling rates are coupled to the higher frequency internal modes of the plant.

It is occasionally possible to decompose a system into several lower order loosely coupled subsystems in which the required sampling rates for each of the signals in any subsystem are approximately the same. If this is the case, a different controller can be independently designed for each subsystem using the standard methods of single-rate controller design. The collection of these independent controllers can be thought of as a simple form of multirate system controller.

If the requirement of loose coupling between the subsystems is not satisfied, the use of independent analysis of each of the individual subsystems is precluded and a different approach must be sought. Although a controller could be designed to run at the rate required by the "fastest" signal, one that handles each signal at its minimum required rate could provide nearly identical system performance with a significant reduction in control computation. Several methods exist for the design of such a multirate controller. These will be reviewed and expanded in

the following chapters.

This thesis is concerned with developing and extending design methods for multirate controllers that utilize modern state space representations of the system. Using these methods, an engineer should be able to obtain a "best" design for any particular situation.

1.2 Scope of Thesis

This thesis builds on the results of Amit (1980) and Glasson (1980) in applying state space analysis to the design of an optimal multirate regulator. Although the regulator described in these papers is optimal, its solution requires considerably more computation than the single-rate regulator and its structure is time varying. These two undesirable features will be addressed in the next several chapters. First, the design of an optimal multirate regulator will be reviewed and an improved algorithm for its solution will be developed. The question of regulator structure will be addressed by proposing a variety of suboptimal multirate regulators having simpler structures than their optimal counterpart. The performance of the suboptimal regulators will be compared to that of the optimal regulator and these comparisons will yield insight into how the most practical regulator for any specific situation can be selected.

The regulator structures and design methods proposed in this thesis are applied to two example systems. The steps taken in the design are described in detail and the

performance of the optimal multirate regulator is compared with the performance of each of the suboptimal regulators. These examples quantify the difference in performance between the optimal multirate regulator, the suboptimal multirate regulators, and the fast and slow single-rate regulators.

The optimal multirate estimator will not be considered in this thesis. As in the single-rate case, the multirate regulator and the multirate estimator are duals. As a result, the procedures developed here can be easily modified to apply to the optimal multirate estimator.

1.3 New Results

The major contributions of this thesis are:

1. A new method for the calculation of the optimal periodic feedback gains of a multirate regulator is presented. The underlying ideas are the same as those used by Amit (1980) and Glasson (1980). A different approach has, however, been taken to the use of additional states that hold control values between updates. This new method makes the solution easier to understand and provides a simpler algorithm for the calculation of the optimal feedback gains of the multirate regulator. It is shown that the resulting optimal regulator is periodically time varying and uses the plant state and the held controls in the calculation of all new controls.
2. Three suboptimal multirate regulators of decreasing complexity are proposed and analyzed. These suboptimal

regulators are:

a. The Plant-State-Only Regulator - This regulator uses the plant states only (and not the held controls) in the calculation of all new controls. It is shown that this regulator is periodically time varying and provides controls that are identical to those calculated by the optimal multirate regulator when used in a deterministic system. In a stochastic system, however, it is shown that the plant-state-only regulator provides suboptimal performance.

b. The Averaged-Gains Regulator - In order to further simplify the structure of the multirate regulator, a constant gain regulator that uses the average of the optimal multirate gains in the calculation of all new controls is presented and analyzed.

c. The Separated-Modes Regulator - In the case where coupling between the modes of the plant is small, near optimal performance can be obtained by using single-rate design techniques to obtain independent single-rate regulators for each of the modes of the plant.

Methods of solution are provided for each of these regulators and the relative ease of solution is discussed. The effect of the simpler structure of the suboptimal regulators on system performance is illustrated by example.

3. Three optimal constrained multirate regulators of decreasing complexity are proposed and analyzed. Each of these regulators has a structure that is identical to

one of the suboptimal regulators and, as a result, they provide a quantitative measure of the optimal performance available from each of the structures.

a. The Optimal Plant-State-Only Regulator - This regulator is only allowed to use the plant state (and not the held controls) in the calculation of all new controls. By comparing its performance to the performance of the optimal regulator, conclusions can be drawn as to the importance of the hold states in providing optimal regulation.

b. The Optimal Constant-Gain Regulator - A time invariant structure is used in this regulator to determine the loss in performance resulting from the use of constant gains. In this case, the held control states are still present but are also forced to use a constant gain. The structure of the constant-gain regulator is the same as that of the suboptimal averaged-gains regulator.

c. The Optimal Constant-Gain Plant-State-Only Regulator - The final multirate regulator to be considered uses a constant gain and the plant state in the calculation of all new controls. Its structure is the same as that of the separated-modes regulator.

4. The design procedure for a multirate regulator is discussed in detail and is applied to two example systems. These examples also provide a quantitative illustration of the effect of regulator structure on system performance.

A multirate system design and analysis package called "MULTIRAT" was constructed to implement the algorithms and design procedures presented in this thesis. This package is able to solve the optimal multirate regulator, the optimal constrained regulators, and the suboptimal regulators. It is also able to calculate the open-loop and closed-loop system eigenvalues and the closed-loop system performance for each of the regulators. MULTIRAT was used in the design and analysis of the two example systems.

1.4 Thesis Outline

This thesis can be divided into three major sections. Chapters 1 and 2 provide an introduction to the multirate controller problem. Chapters 3 through 5 present the new results of the thesis and chapters 6 through 9 apply the proposed design methods to examples and draw conclusions.

A review of previous related results is provided in chapter 2. The early methods of frequency domain analysis are reviewed briefly for completeness but will receive no further attention. Recent results using state space analysis techniques are discussed in somewhat greater detail. Also in chapter 2, the optimal multirate control system is introduced and the notation used in its description is given. A state space formulation is used in introducing the multirate control system and all further analysis will use state space representations exclusively.

Chapter 3 discusses the optimal multirate regulator problem and develops a new method for its solution. This

method is a modified form of those published in recent papers.

Chapter 4 introduces the three optimal constrained regulators and provides a numerical method for their solution.

Chapter 5 introduces the three suboptimal regulators and develops methods for their solution.

The procedures and mathematical tools needed to analyze a multirate system are provided in chapter 6.

Two example systems are used to illustrate the application of multirate regulator design in chapters 7 and 8. The suboptimal regulators proposed in the earlier chapters are solved for these systems and their performance is compared to that of the optimal multirate regulator.

Chapter 9 summarized the results of the preceding chapters and gives recommendations for further areas of research.

Chapter 2

THE MULTIRATE SYSTEM

2.1 Introduction

This chapter introduces the multirate control system and provides a brief review of its history. The early papers that used classical control theory are outlined in the section on frequency domain techniques. Developments of the theory using state space representations are outlined in the section devoted to time domain techniques. A more detailed review of some of the earlier work in the area of multirate control can be found in Glasson (1983).

The notation and mathematical representations for the multirate system are also introduced in this chapter. These include both the discrete-time and continuous-time descriptions of the plant and optimal performance index. In addition, the various possible sampling schemes in a multirate system are discussed and notation to represent them is developed.

2.2 Frequency Domain Techniques

The first significant contribution to the analysis of multirate systems is attributed to Sklansky (1955). Although Sklansky's analysis method, the "frequency decomposition technique", was initially concerned with the intersample behaviour of single-rate systems, it could also be used in the analysis of multirate systems. A similar technique known as the "switch decomposition technique" was introduced by Kranc (1956). The "switch decomposition technique" provided

a method for representing a multirate system as an equivalent single-rate system. This technique is similar to that applied by Glasson (1980) and Amit (1980) and further developed in this thesis in the time domain analysis of multirate systems. A few years later, these two early decomposition techniques were presented in a textbook by Ragazzini and Franklin (1959).

During the 1960's and early 1970's little attention was paid to the multirate control problem and the few papers that were published were primarily concerned with the extension of existing techniques. Notable contributions during this period are those of Coffey and Williams (1966) and Jury (1967). More recent contributions include those of Whitbeck (1978,1980).

These early methods of analysis were frequency domain techniques and, although easy to understand, they result in system dimensionality growth that becomes unmanageable for higher order systems. Also, like most frequency domain techniques, they are not easily applied to the analysis of multi-input, multi-output, or time varying plants.

2.3 Time Domain Techniques

Multirate system design and analysis using modern time domain techniques is a young and exciting area of research. In the mid 1970's, Barry (1975) described the design of a multirate regulator and showed that its performance is superior to a single-rate regulator having the same "slow" sample rate. The optimal multirate estimator was considered

by Sandell (1977). In his paper, Sandell showed that under appropriate assumptions, the multirate estimator reaches a periodic steady state. He also formulated controllability and observability conditions for the multirate system.

The earlier papers of Barry (1975) and Sandell (1977) set the stage for the detailed works of Glasson (1980) and Amit (1980). The TASC report published by Glasson and the doctoral thesis published by Amit each dealt with the complete problem of the analysis and synthesis of the multirate estimator and the multirate regulator. Each of these works formulated, and then illustrated by example, the design of an optimal multirate estimator and regulator. Also, in each of these works, the performance of the optimal multirate control system was quantitatively illustrated by example.

The results presented in this thesis provide an extension to the works of Glasson and Amit. Many of the techniques and much of the notation has been borrowed from those works.

2.4 Problem Statement

A multirate system is a multiple input, multiple output, digitally controlled system in which each of the individual outputs can be sampled at different rates, and each of the individual inputs can be updated at different rates. The major concern of this thesis is the multirate regulator which calculates the different control input values based on perfect knowledge of the system state.

Duality of the regulator and estimator problems guarantees that the results obtained for the multirate regulator will apply to the multirate estimator as well.

The structure of the multirate regulator for a system having two control inputs is shown in Figure 2.1. The "fast" control input, u_f, is calculated and updated every T seconds while the "slow" control input, u_s, is calculated and updated every kT seconds and held at its previous value at all other "fast" control updates.

The state equation for the continuous-time plant is given by

$$\dot{x}(t) = F(t)x(t) + G(t)u(t) + G_v(t)w(t) \qquad (2.4.1)$$

where the state vector, $x(t)$, the control vector, $u(t)$, and the process noise vector, $w(t)$, belong to n, m and r dimensional real vector spaces respectively. It is assumed that the process noise is white, with zero mean, and a Gaussian distribution. For simplicity, it is further assumed that the state propagation matrix, $F(t)$, the control distribution matrix, $G(t)$, and the disturbance distribution matrix, $G_v(t)$, are time invariant.

Although the plants to be considered are linear and time invariant, the analysis and design of multirate systems is easily generalized to periodically time varying plants with periods equal to the period of the sampling scheme.

The discrete-time form of the state equation is given by

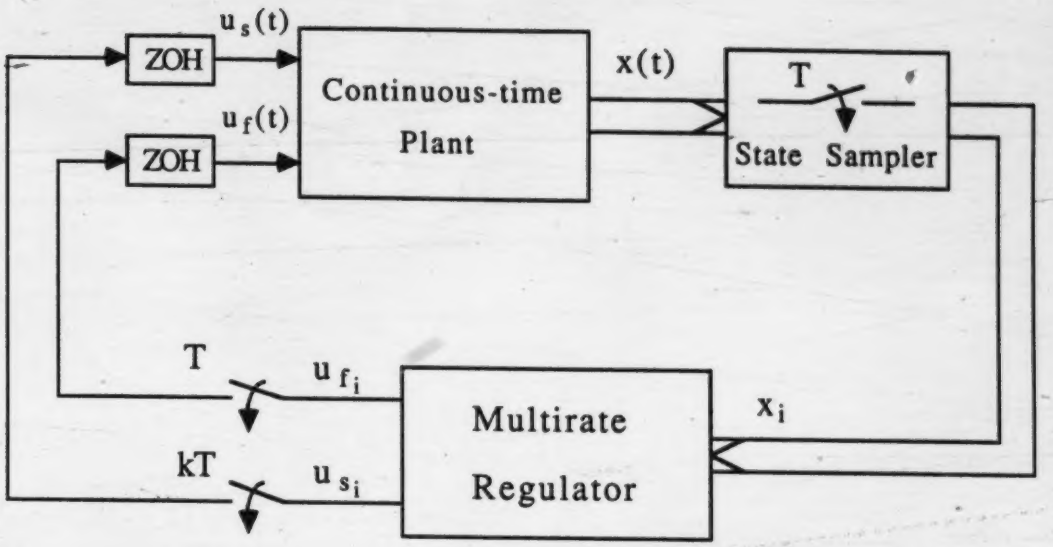

Figure 2.1 - Multirate Regulator for System Having Two Control Inputs

$$x_{i+1} = \phi x_i + \Gamma u_i + w_i \qquad (2.4.2)$$

$$t_{i+1} - t_i = T \qquad (2.4.3)$$

where x_i, u_i and w_i are of the same dimension as $x(t)$, $u(t)$, and $w(t)$ respectively. As a result of the assumption that F and G are constant, the matrices ϕ and Γ will also be constant. The amount of time between discrete time instants, T, will be refered to as one "cycle". Note that while x_i and u_i are just the sampled versions of $x(t)$ and $u(t)$, w_i is the integrated effect of the process noise over the i-th cycle.

It is assumed that the regulator has perfect knowledge of the state, x_i, at time t_i. With this knowledge, the

multirate regulator calculates the control vectors, u_i, with each element possibly being updated at a different rate. These control inputs are provided to the continuous-time plant through zero-order hold (ZOH) circuits.

The optimal multirate regulator will be designed to calculate the control vectors, u_i, which minimize the quadratic cost

$$J = \lim_{t\to\infty} \frac{1}{t} E\left\{\int_0^t (x(\tau)^T Q_A x(\tau) + u(\tau)^T R_A u(\tau))d\tau\right\} \quad (2.4.4)$$

where the continuous-time state weighting matrix, Q_A, and the continuous-time control weighting matrix, R_A, are given. It is also required that Q_A be positive semidefinite and R_A be positive definite.

In terms of the discrete-time variables, this cost can be written as

$$J = \lim_{N\to\infty} \frac{1}{N} E\left\{\sum_{i=0}^{N} [x_i^T \; u_i^T]\begin{bmatrix} Q & M \\ M^T & R \end{bmatrix}\begin{bmatrix} x_i \\ u_i \end{bmatrix}\right\} \quad (2.4.5)$$

where the discrete state weighting matrix, Q, the discrete control weighting matrix, R, and the discrete cross weighting matrix, M, can be calculated from Q_A and R_A. (The details of these calculations are given in Chapter 6.)

2.5 The Multirate Sampling Scheme

The multirate regulator can be thought of as a single-rate regulator that is restricted by specifying that some of its controls must hold their previous values at certain time instants. The specification of which controls are allowed to change and which are not allowed to change at each time instant is inherent in the specification of the sampling rates of each of the control inputs.

If the ratios of the sampling rates of all control inputs are rational numbers, the sampling scheme will repeat itself after a constant number of cycles. This repetition period will be designated as k cycles and it can be mathematically expressed as the least common multiple of all of the sampling periods.

$$kT = \text{LCM(all of the individual sampling periods)} \quad (2.5.1)$$

The amount of time after which the sampling scheme repeats itself (kT), will be called the "control period".

To formally describe the control updates, group the control vectors over one control period into a matrix.

$$[u_n \; u_{n+1} \; \ldots \; u_{n+k-1}] \quad (2.5.2)$$

The sampling scheme can now be represented by an "update" matrix of the same dimension.

$$UP = \begin{matrix} \\ \text{control} \\ \text{input} \end{matrix} \begin{matrix} \overset{\text{time}}{\rightarrow\rightarrow\rightarrow\rightarrow} \\ \downarrow \\ \downarrow \\ \downarrow \end{matrix} \begin{bmatrix} \\ \bullet \\ \\ \end{bmatrix} \qquad \text{DIM}(m \times k) \qquad (2.5.3)$$

where

$UP(i,j) = 1$

if the i-th control input can change at the j-th cycle in the control period.

$UP(i,j) = 0$

if the i-th control input must hold its previous value at the j-th cycle in the control period.

Example 2.1:

For a system having two control inputs with the first control input updated at three times the rate of the second control input, the update matrix would be given by

$$m = 2 \qquad k = 3$$

$$UP = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}. \qquad (2.5.4)$$

2.6 Summary

In this chapter the state-of-the-art in multirate controller design has been outlined. The frequency domain techniques were briefly mentioned but, due to their inherent limitations, they will not be explored further. The use of

time domain techniques in the design and analysis of multirate systems is relatively recent. Of the few papers that have been published in this area, those of Glasson (1980) and Amit (1980) are central to this thesis. Their methods and variations on their methods will be discussed in detail in subsequent chapters.

The continuous-time plant and the optimal multirate regulator were also introduced in this chapter. It was stated that the optimal multirate regulator will be designed to minimize a quadratic performance index that weights the states and controls of the continuous-time plant. The discrete-time form of the state equation and cost functional were given. The conditions for a periodic sampling scheme were reviewed and a notational method for representing the sampling scheme was developed.

Chapter 3

THE OPTIMAL MULTIRATE REGULATOR

3.1 Introduction

In this chapter, the solution of the optimal multirate regulator is discussed in detail. In order to obtain the feedback gains, the time varying multirate system is first transformed into an equivalent time invariant system. The optimal regulator for this system is solved using standard methods. In order to solve the optimal regulator for the original time varying system, a cycle-by-cycle representation for the multirate state and cost equations is developed. Using this time varying representation, the Riccati equation is solved at each cycle by propagating it over the control period. From the Riccati equation solution at each cycle, the feedback gains can be directly calculated and the solution is complete. In order to keep the solution from becoming too cluttered, some of the mathematical details are delayed until chapter 6.

3.2 Problem Statement

Consider the discrete-time system

$$x_{i+1} = \phi x_i + \Gamma u_i + w_i \qquad (3.2.1)$$

$$t_{i+1} - t_i = T \qquad (3.2.2)$$

where the state vector, x_i, the control vector, u_i, and the disturbance vector, w_i, belong to n, m, and r dimensional

real vector spaces respectively. For simplicity, it is assumed that the state transition matrix, ϕ, and the control distribution matrix, Γ, are constant.

It is also assumed that the process noise giving rise to w_i has a correlation time that is much shorter than the sample period. With this, the w_i can be assumed to be uncorrelated. They are described by

$$w_i \sim N[O, W_d]. \tag{3.2.3}$$

That is, w_i is Gaussian with zero mean and covariance W_d.

The multirate regulator will be designed to calculate the control vectors, u_i, that minimize the quadratic cost

$$J = \lim_{N\to\infty} \frac{1}{N} E\left\{ \sum_{i=0}^{N} [x_i^T \; u_i^T] \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix} \right\} \tag{3.2.4}$$

where the matrices Q, M, and R are given.

In addition, the system will be further restricted by specifying that some of the controls must hold their previous values at certain time instants. The specification of which controls are allowed to change and which are not allowed to change will be given by the "m by k" update matrix, UP, introduced in Chapter 2.

3.3 Method of Solution

The multirate regulator and its associated plant is a periodically time varying system with a period equal to the control period of the sampling scheme. As a result, the

"Linear Quadratic Gaussian" (LQG) formulations for time varying systems (see Levis and Dorato, 1971) can be applied to the multirate regulator problem as stated in the previous section. It has been shown by Sandell (1977) that the Riccati equation for such a system is periodic and its solution will reach a periodic steady state after sufficient propagation. This approach was used by Glasson (1980) to find the optimal regulator gains. Due to the large amount of computation required to propagate the Riccati equation to steady state, this method is undesirable and will not be used here.

A cleaner solution would be to apply the eigenvector methods of Franklin and Powell (1980) or Laub (1979) to obtain the solution of the Riccati equation. Unfortunately, these methods are applicable only to time invariant systems and cannot be directly used to design the multirate regulator. Amit (1980) approached this problem by converting the multirate system into an equivalent, time invariant, "single-rate system". The same approach is taken in this thesis. Note, however, that the method of finding the equivalent, time invariant, system presented here is different from that proposed by Amit.

The propagation of the state during one control period is represented in Figure 3.1. At the start of the control period, t_n, the state is x_n and the control is u_n. This state will propagate to the state x_{n+1} according to

$$x_{n+1} = \phi x_n + \Gamma u_n. \qquad (3.3.1)$$

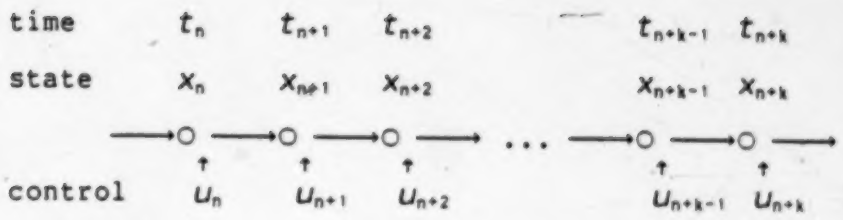

Figure 3.1 - State Propagation Over One Control Period

Likewise, all states up to the end of the control period will propagate according to the same equation. † Collecting the effects of each of the control inputs during the control period, the state at the end of the control period can be related to the state at the beginning of the control period and all of the controls applied during the control period.

$$x_{n+k} = f(\phi,\Gamma;x_n,u_n,u_{n+1},\ldots,u_{n+k-1}) \qquad (3.3.2)$$

It will be shown in a later section that this relationship is of the form

$$x_{n+k} = \Phi x_n + \Gamma_k U_n \qquad (3.3.3)$$

where U_n contains all of the controls applied during the

---

†Note that the deterministic state equation is being used here even though the system is subject to disturbance inputs. This is justified since it is known that the feedback link in an optimal regulator is independent of the properties of the disturbance inputs. (See, for example, Kwakernaak and Sivan, 1973, pp 259-269.)

control period and Φ and Γ_1 are functions of ϕ and Γ.

This method of obtaining a time invariant representation for a periodically time varying system has been used in the literature (e.g. Meyer and Burrus, 1975).

A steady state optimal control can be obtained by applying a method such as that described by Laub (1979). The resulting regulator will be of the form

$$U_n = Kx_n \qquad (3.3.4)$$

Here each of the controls during the control period have been specified (through U_n) in terms of the state at the beginning of the control period, x_n. The desired solution, however, is of the state feedback form

$$u_{n+i} = c_i x_{n+i} \qquad (0 \le i < k) \qquad (3.3.5)$$

where the controls at each cycle use the latest available information, i.e., the state at the current cycle and not the state at the beginning of the control period. Although these two solutions are equivalent in the deterministic case, they differ in the stochastic case, and it is expected that superior control can be obtained by using current state information.

The desired form of the solution will be obtained by noting that the Riccati equation for the equivalent time invariant system is equal to the solution of the Riccati equation for the periodically time varying system at the beginning of the control period. Using the solution of the

equivalent time invariant Riccati equation, the periodic Riccati equation can be propagated over one control period and the feedback gains calculated.

3.4 The Multistage System

Before discussing the multirate system, the method used to convert it to time invariant form will be illustrated on a single-rate system. The method basically involves using one control period rather than one cycle for the discrete time base.

Consider again the several stages of the system state propagation represented by Figure 3.1. For this system, the state at time t_{n+k} can be expressed in terms of the state at time t_n and the controls applied to the system between the times t_n and t_{n+k}.

$$x_{n+k} = f(\phi,\Gamma;x_n,u_n,u_{n+1},\ldots,u_{n+k-1}) \qquad (3.4.1)$$

where t_n is chosen to be the beginning of a control period and t_{n+k} will then be the beginning of the next control period. Assuming a fixed value for k, an expression for x_{n+k} can be obtained as follows.

$$\begin{aligned}
x_{n+1} &= \phi x_n + \Gamma u_n \qquad (3.4.2)\\
x_{n+2} &= \phi x_{n+1} + \Gamma u_{n+1} = \phi^2 x_n + \phi\Gamma u_n + \Gamma u_{n+1}\\
x_{n+3} &= \phi x_{n+2} + \Gamma u_{n+2} = \phi^3 x_n + \phi^2\Gamma u_n + \phi\Gamma u_{n+1} + \Gamma u_{n+2}\\
&\vdots\\
x_{n+k} &= \phi x_{n+k-1} + \Gamma u_{n+k-1} = \phi^k x_n + \phi^{k-1}\Gamma u_n + \ldots + \Gamma u_{n+k-1}
\end{aligned}$$

Define the augmented control vector, U_n, to contain all control vectors over one control period.

$$U_n = \begin{bmatrix} u_n \\ u_{n+1} \\ \cdots \\ u_{n+k-1} \end{bmatrix} \qquad \text{DIM}((k*m)\times(1)) \qquad (3.4.3)$$

Then write

$$x_{n+i} = \phi^i x_n + [\phi^{i-1}\Gamma \ \phi^{i-2}\Gamma \ \ldots \ \phi\Gamma \ \Gamma \ 0 \ \ldots \ 0]U_n. \qquad (3.4.4)$$

Further define

$$\Gamma_i = [\phi^{i-1}\Gamma \ \phi^{i-2}\Gamma \ \ldots \ \phi\Gamma \ \Gamma \ 0 \ \ldots \ 0] \qquad \text{DIM}(n\times(m*k)) \qquad (3.4.5)$$

and write

$$x_{n+i} = \phi^i x_n + \Gamma_i U_n \qquad (0<i\leq k). \qquad (3.4.6)$$

The equation for the transition of the state from the beginning of one control period to the beginning of the next is then given by

$$x_{(j+1)} = \phi^k x_{(j)} + \Gamma_k U_{(j)} \qquad (3.4.7)$$

where the bracketed subscripts on $x_{(j)}$ and $U_{(j)}$ indicate that the states and augmented controls at the beginning of each control period are the only ones of interest.

Now the quadratic cost must also be written with the control period as its time base. Observe that

$$
\begin{aligned}
U_n &= [I\ 0\ 0\ \ldots\ 0]U_n = \theta_0 U_n \\
U_{n+1} &= [0\ I\ 0\ \ldots\ 0]U_n = \theta_1 U_n \\
&\vdots \\
U_{n+k-1} &= [0\ 0\ \ldots\ 0\ I]U_n = \theta_{k-1} U_n
\end{aligned}
\tag{3.4.8}
$$

or

$$
U_{n+i} = \theta_i U_n \qquad (0 \le i < k). \tag{3.4.9}
$$

The state and control at any time during the control period can be written in terms of the initial state and the augmented control.

$$
\begin{bmatrix} x_{n+i} \\ U_{n+i} \end{bmatrix} = \begin{bmatrix} \phi^i & \Gamma_i \\ 0 & \theta_i \end{bmatrix} \begin{bmatrix} x_n \\ U_n \end{bmatrix} = T_i \begin{bmatrix} x_n \\ U_n \end{bmatrix} \tag{3.4.10}
$$

where T_i has been defined by

$$
T_i = \begin{bmatrix} \phi^i & \Gamma_i \\ 0 & \theta_i \end{bmatrix} = \begin{bmatrix} \phi^i & \phi^{i-1}\Gamma & \phi^{i-2}\Gamma & \ldots & \phi\Gamma & \Gamma & 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & \ldots & 0 & 0 & I & 0 & \ldots & 0 \end{bmatrix} \tag{3.4.11}
$$

$$
\text{DIM}((n+m)\times(n+k*m)).
$$

With these relationships, the cost over one control period can be expressed in terms of the initial state and the controls applied during the control period. Define the cost accumulated over one control period as j_{cp}. Then using the quadratic cost,

$$
J = \lim_{N\to\infty} \frac{1}{N} E\left\{ \sum_{i=0}^{N} [x_i^T\ U_i^T] \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} \begin{bmatrix} x_i \\ U_i \end{bmatrix} \right\}, \tag{3.4.12}
$$

$$J = \lim_{N\to\infty} \frac{1}{N} E\left\{ \sum_{cp=0}^{N/k} j_{cp} \right\} \qquad (3.4.13)$$

with

$$j_{cp} = \sum_{i=0}^{k-1} [x_i^T \; u_i^T] \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix} = \sum_{j=0}^{k-1} j_i \qquad (3.4.14)$$

where j_i is the cost over one cycle and the indices on x_i, u_i, and j_i are referenced to the beginning of the cp-th control period. Using the definition of T_i,

$$j_i = [x_n^T \; u_n^T] \; T_i^T \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} T_i \begin{bmatrix} x_n \\ u_n \end{bmatrix}. \qquad (3.4.15)$$

At steady state j_{cp} will be constant and the cost can be written as

$$J = \frac{1}{k} E\left\{ j_{cp} \right\}. \qquad (3.4.16)$$

This is the expected value of the cost over one cycle at steady state.

Using these relationships the optimal discrete-time control problem for a multistage system can now be stated.

OPTIMAL MULTISTAGE CONTROL PROBLEM

Given the state equation

$$x_{(j+1)} = \phi^k x_{(j)} + \Gamma_k U_{(j)} \tag{3.4.17}$$

find the controls $U_{(j)}$ to minimize the quadratic cost

$$J = \frac{1}{k} E\left\{ \begin{bmatrix} x_{(j)}^T & U_{(j)}^T \end{bmatrix} T \begin{bmatrix} x_{(j)} \\ U_{(j)} \end{bmatrix} \right\} \tag{3.4.18}$$

where

$$T = \sum_{i=0}^{k-1} T_i^T \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} T_i . \tag{3.4.19}$$

These equations describe a multistage system that is equivalent to the original single stage system. The standard LQG design methodology can be directly applied to this system. In the next section, it will be shown that the structure of the multistage system can be used to obtain a time invariant representation for a multirate system.

Note that when transforming a system to an equivalent multistage system, the dimensions of the control distribution and control weighting matrices can become quite

large if k is large. For this reason, it is detrimental to transform a regular time invariant system into an equivalent multistage system for design or analysis.

3.5 Conversion to Time Invariant Form

In a multirate system, each control input can be updated at a different rate. The control vector u_i at time t_i can have some elements that are new while others hold the same values they had at time t_{i-1}. As discussed in an earlier section, the exact specification of which elements of each control vector are new, and which are held, can be easily determined from the update rates of each of the control inputs. If the control vectors during the control period are grouped into the matrix

$$[u_n \; u_{n+1} \; \ldots \; u_{n+k-1}] \qquad \text{DIM}(n \times k), \qquad (3.5.1)$$

the first index of each element will indicate the control input number and the second index will indicate the time in the control period. Which elements are to be held and which are to be updated can be specified by an update matrix.

$$UP = [up_{i,j}] \qquad (3.5.2)$$

where

$up_{i,j} = 1$

if control input u_i at time t_{j-1} (from the beginning of the control period) is new.

$up_{i,j} = 0$

if control input u_i at time t_{j-1} is held from time t_{j-2}.

As in the previous section, the controls over the control period can be written as an augmented control vector.

$$U_n = \begin{bmatrix} u_n \\ u_{n+1} \\ \cdots \\ u_{n+k-1} \end{bmatrix} \qquad \text{DIM}((k*m)\times(1)) \qquad (3.5.3)$$

While all elements of the augmented control vector for the multistage system could be arbitrarily specified, several elements of the augmented control vector for the multirate system are forced to be identical. In order to utilize the methods of optimal regulator design, these elements must be eliminated. After the elimination process is complete, the control vector will only contain elements that can be arbitrarily specified. This vector, U_m, will be called the "reduced control vector" of the multirate system. The dimensions of U_m are "l by 1", with l being the total number of new control inputs applied during one control period.

It will be shown that since the augmented control vector U contains the same information as the reduced control vector U_m, they can be related by the transformation

$$U = EU_m \qquad (3.5.4)$$

where E is an "$m*k$ by 1" matrix. The purpose of E is to mathematically express the fact that some controls need to be held.

In the optimal multistage control problem, U appears only in the terms $\Gamma_i U$ or $\theta_i U$. As a result, a better understanding of the features of the E transformation can be obtained by considering products of the form AU where A is an arbitrary matrix. These features are best illustrated by example.

Example 3.1:

Consider the two control, three cycle system of Example 2.1.

$$m = 2 \qquad k = 3$$

$$UP = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{3.5.5}$$

The augmented control vector for this system can be written as

$$U = \begin{bmatrix} u_{01} \\ u_{02} \\ - \\ u_{11} \\ u_{12} \\ - \\ u_{21} \\ u_{22} \end{bmatrix} \tag{3.5.6}$$

where the first index on each u represents the cycle number in the control period and the second index represents the control input number.

The arbitrary matrix A is written in columnar form as

$$AU = [A_1 | A_2 | A_3 | A_4 | A_5 | A_6]\, U \qquad (3.5.7)$$
$$= A_1U_{01} + A_2U_{02} + A_3U_{11} + A_4U_{12} + A_5U_{21} + A_6U_{22}$$

From the update matrix, it can be seen that

$$U_{22} = U_{12} = U_{02} \qquad (3.5.8)$$

so that

$$AU = A_1U_{01} + (A_2 + A_4 + A_6)U_{02} + A_3U_{11} + A_5U_{21} \qquad (3.5.9)$$

where the control inputs U_{01}, U_{02}, U_{11}, and U_{21} can be arbitrarily specified. AU can be written in terms of the reduced control vector, U_m, as

$$AU = [A_1 | (A_2 + A_4 + A_6) | A_3 | A_5] \begin{bmatrix} U_{01} \\ U_{02} \\ U_{11} \\ U_{21} \end{bmatrix} \qquad (3.5.10)$$

or

$$AU = A'U_m. \qquad (3.5.11)$$

From the above example, it can be seen that if a transformation matrix exists such that

$$U = EU_m \qquad (3.5.12)$$

then

$$AE = A'. \qquad (3.5.13)$$

In other words, postmultiplication by the transformation matrix causes the columns of A that are multiplied by held controls to be added back into the columns of A that multiply those controls when they were initially calculated. The columns of A that were multiplied by held controls are then discarded by the transformation.

This type of transformation can be obtained by performing the equivalent column summations and eliminations on an identity matrix of dimension "$(m*k)$ by $(m*k)$". (See also Noble and Daniel, 1977.)

For the above example, E is given by

$$E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \qquad (3.5.14)$$

It can be seen from the example that UP uniquely determines E and E uniquely determines UP. Thus, E can be thought of as another representation for the sampling scheme. †

---

†An algorithm to calculate E has been coded and is presently being used in the MULTIRAT software package.

3.6 The Solution of the Optimal Multirate Regulator

Using the multistage state equation

$$x_{(j+1)} = \phi^k x_{(j)} + \Gamma_k U_{(j)} \qquad (3.6.1)$$

and the transformation equation

$$U_{(j)} = E U_{m(j)}, \qquad (3.6.2)$$

the equivalent time invariant form of the state equation for the multirate system can be written.

$$x_{(j+1)} = \phi^k x_{(j)} + \Gamma_k E U_{m(j)} \qquad (3.6.3)$$

where, again, bracketed subscripts have been used to indicate that the states and augmented controls at the beginning of each control period are the only ones of interest.

In order to write an expression for the cost accumulated over one control period in terms of $x_{(j)}$ and $U_{m(j)}$, use

$$\begin{bmatrix} x_{(j)} \\ U_{(j)} \end{bmatrix} = E_a \begin{bmatrix} x_{(j)} \\ U_{m(j)} \end{bmatrix} \qquad (3.6.4)$$

where the augmented E matrix, E_a, is defined by

$$E_a = \begin{bmatrix} I & 0 \\ 0 & E \end{bmatrix} \qquad \text{DIM}((n+k*m)\times(n+1)). \qquad (3.6.5)$$

Using these relationships, the equivalent time invariant form of the optimal multirate regulator problem can now be stated.

EQUIVALENT TIME INVARIANT FORM OF THE OPTIMAL MULTIRATE REGULATOR PROBLEM

Given the state equation

$$x_{(j+1)} = \phi^k x_{(j)} + \Gamma_k E u_{m(j)} \tag{3.6.6}$$

find the controls $u_{m(j)}$ to minimize the quadratic cost

$$J = \frac{1}{k} E\left[[x_{(j)}^T \; u_{m(j)}^T] \; T \begin{bmatrix} x_{(j)} \\ u_{m(j)} \end{bmatrix} \right] \tag{3.6.7}$$

where

$$T = \sum_{i=0}^{k-1} E_a^T T_1^T \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} T_1 E_a . \tag{3.6.8}$$

These equations describe a time invariant system that is equivalent to the original multirate system. As in the case of the multistage system, the standard LQG design methodology can be directly applied to this system.

3.7 Calculation of the Optimal Feedback Gains

Solution of the equivalent time invariant system using the LQG formulations gives a regulator that specifies each of the controls in the control period in terms of the state at the beginning of the control period. This form is to be expected since the state at the beginning of each control period was represented as a function of the state at the beginning of the previous control period and all controls applied over the control period. This representation, however, has not taken the plant disturbances into account. Due to the presence of these disturbances, better system performance should be obtainable by calculating the control vectors at each cycle using the present state rather than the state at the beginning of the control period. The desired control vectors would be of the form

$$U_i = C_i X_i \qquad (3.7.1)$$

where U_i, X_i, and C_i are referenced to the beginning of the control period.

This state feedback law can be obtained by noting the similarities between the equivalent time invariant form of the multirate system and the periodically time varying form of the multirate system. The state of the equivalent time invariant system is identical to the state of the time varying multirate system at the beginning of each control period. The state propagation equation for the equivalent time invariant system also describes the state propagation of the multirate system from the beginning of one control

period to the beginning of the next. Likewise, the cost of control of the equivalent time invariant system, although written in a different form, will have the same values as that of the multirate system. With these equivalences, it seems reasonable to expect that the solution of the Riccati equation of the equivalent time invariant system should equal that of the multirate system at the beginning of each control period. This, in fact, turns out to be true and the details of its proof can be found in Amit (1980).

With knowledge of the Riccati equation solution at the beginning of the control period, the solution for each cycle in the control period can be obtained by propagating the Riccati equation backwards to the beginning of the previous control period. If the initial value was, in fact, the steady state solution of the Riccati equation, the value obtained after propagation over one control period should be the same.

From the solution of the Riccati equation at each cycle, the optimal feedback gains can be directly calculated. The details of these well known calculations are delayed until chapter 6.

3.8 Structure of the Optimal Multirate Regulator

In order to propagate the Riccati equation over the control period, the structure of the multirate system must be examined in greater detail. The multirate system has the unique characteristic that some of the controls at each cycle are new while others are held from the previous cycle.

The held controls represent information from the past that will affect the future and, as a result, they must be stored or remembered. From this point of view, it is appropriate to think of the held controls as additional states. These states have no dynamics (they maintain a constant value) and appear and disappear as needed. A variable dimension vector containing the plant states and the held controls will be used to represent the state of the complete system. It will be called the "augmented state vector". In order to represent the propagation of the augmented state, the state transition matrix, the control distribution matrix, and the state, control, and cross weighting matrices must be appropriately modified. The form of the modified system matrices can be seen by writing

$$x_{i+1} = \phi x_i + \Gamma_{h_i} u_{h_i} + \Gamma_{n_i} u_{n_i} \qquad (3.8.1)$$

where u_{h_i} are the held controls and u_{n_i} are the new controls. Γ_{h_i} contains the columns of Γ that multiply the held controls and Γ_{n_i} contains the columns of Γ that multiply the new controls.

The propagation of the hold states can be represented as

$$u_{h_{i+1}} = \theta_{h_i} u_{h_i} + \theta_{n_i} u_{n_i} \qquad (3.8.2)$$

where θ_{h_i} "picks up" any previously held controls that are to be held for another cycle, and θ_{n_i} "picks up" any new controls that are to be held into the next cycle.

θ_{h_i} can be formed by eliminating appropriate rows and columns of an "m by m" identity matrix. Columns corresponding to all new controls in the current cycle are eliminated and rows corresponding to all new controls in the next cycle are eliminated.

Likewise, θ_{n_i} can be formed by eliminating appropriate rows and columns of an "m by m" identity matrix. Columns corresponding to all held controls in the current cycle are eliminated and rows corresponding to all new controls in the next cycle are eliminated.

Now the augmented state propagation equation can be written as

$$\begin{bmatrix} x_{i+1} \\ u_{h_{i+1}} \end{bmatrix} = \begin{bmatrix} \phi & \Gamma_{h_i} \\ 0 & \theta_{h_i} \end{bmatrix} \begin{bmatrix} x_i \\ u_{h_i} \end{bmatrix} + \begin{bmatrix} \Gamma_{n_i} \\ \theta_{n_i} \end{bmatrix} u_{n_i} \tag{3.8.3}$$

or

$$x_{a_{i+1}} = \phi_{a_i} x_{a_i} + \Gamma_{a_i} u_{n_i} \tag{3.8.4}$$

where x_a is the augmented state vector, ϕ_a is the augmented state transition matrix, and Γ_a is the augmented control distribution matrix.

The augmented state, control, and cross weighting matrices (Q_a, R_a, and M_a, respectively) are obtained by moving appropriate rows and columns between the state, control, and cross weighting matrices given in the problem statement (Q, R, and M, respectively). The necessary modifications are easily seen by noting that the held

controls that were previously weighted by R are now part of the state and must be weighted by Q_a. This is accomplished by moving the rows and columns of R, M, and M^T, that weight held controls into the rows and columns of Q_a that weight the held controls. The transformation can be represented as

$$\begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} \rightarrow \begin{bmatrix} Q_{a_i} & M_{a_i} \\ M_{a_i} & R_{a_i} \end{bmatrix}. \tag{3.8.5}$$

Note that since the number of hold states varies from one cycle to the next, the augmented weighting matrices will be periodically time varying.

With this newly defined set of augmented matrices, the multirate system can be represented on a cycle by cycle basis as

$$x_{a_{i+1}} = \phi_{a_i} x_{a_i} + \Gamma_{a_i} u_{n_i} \tag{3.8.6}$$

with

$$J_i = [x_{a_i}^T \; u_{n_i}^T] \begin{bmatrix} Q_{a_i} & M_{a_i} \\ M_{a_i} & R_{a_i} \end{bmatrix} \begin{bmatrix} x_{a_i} \\ u_{n_i} \end{bmatrix}. \tag{3.8.7}$$

Using these augmented matrices, the Riccati equation can be propagated over the control period. The feedback gains at each cycle can then be calculated directly by using the standard LQG formulations.

Another unique and interesting characteristic of the optimal multirate regulator results from the fact that the held controls can be thought of as additional states. With the definition of the augmented state, the regulator has the

option of using the values of the held controls in its calculation of the new controls. The multirate regulator does, in fact, use this additional information to calculate the optimal controls at each cycle.

A heuristic understanding of the value of this additional information can be obtained by considering the selection of new control values from one cycle to the next. In the regulator problem, controls must be found that drive the plant state to zero. At the same time, the state is also being continually perturbed by process noise. Consider the regulator at the beginning of one particular cycle. In order to drive the state to zero, the state itself must be known. The held controls, however, also affect the state propagation. Depending on the state at the beginning of the previous cycle, the held controls will be helping (or hindering) state regulation by varying amounts. Without knowledge of the held controls, they appear as unknown external disturbances. Knowledge of the held controls, on the other hand, allows the regulator to optimally calculate new controls to assist (or possibly counteract) the held controls in driving the state to zero.

3.9 Summary

In this chapter, the solution of the optimal multirate regulator has been discussed in detail. The steady state solution of the periodic Riccati equation was obtained by transforming the multirate system into an equivalent time invariant form and then using one of the popular eigenvector methods. A time varying cycle-by-cycle representation for

the multirate state and cost equations was then developed. Using this time varying representation, the Riccati equation was solved at each cycle by propagating the steady state solution over the control period. From the Riccati equation solution at each cycle, the feedback gains can be directly calculated.

Two interesting features of the optimal multirate regulator are its use of held controls in the calculation of new controls and its use of periodically time varying gains. These characteristics will be examined in somewhat more detail in the following chapters.

Chapter 4

THE OPTIMAL CONSTRAINED MULTIRATE REGULATOR

4.1 Introduction

Control system design involves the examination of a variety of cost/performance tradeoffs. A system that is mathematically optimal can easily be economically unfeasible and the designer is often forced to find a simpler solution to the problem.

This chapter examines three multirate regulators that have simpler structures than the optimal multirate regulator. They are obtained by placing constraints on the structure of the optimal regulator. Each of the regulators is designed to minimize the same cost functional used for the optimal multirate regulator and, for this reason, they are also called optimal. As a result of the constraint on their structure, however, the cost of control will be greater than for the optimal unconstrained regulator.

The structure of the unconstrained regulator had two unique features. The feedback gains were periodically time varying and the held controls were used in the calculation of the new controls. The first two regulators considered in this chapter are obtained by constraining each of these features individually. The third regulator is obtained by applying both constraints simultaneously.

The methods used in this chapter are similar to those proposed by Broussard (1984). In his paper, Broussard considered the use of plant outputs in the calculation of

control inputs for a multirate system. The gains calculated by his algorithm were constant over the control period.

4.2 Method of Solution

The solution of the optimal multirate regulator was obtained by applying the LQG formulations. These formulations provide an analytic method for the calculation of the optimal feedback gains. Unfortunately, as soon as constraints are placed on the feedback gains, the LQG formulations are no longer applicable and an alternate method of solution must be sought. Because of the complexity of the multirate system, a numerical approach was taken.

Consider again the state and control equations for the multirate system obtained in section 3.8.

$$x_{a_{i+1}} = \phi_{a_i} x_{a_i} + \Gamma_{a_i} u_{n_i} + w_{a_i} \qquad (4.2.1)$$

where x_{a_i} is the augmented state, ϕ_{a_i} is the augmented state transition matrix, Γ_{a_i} is the augmented control distribution matrix, u_{n_i} are the new control inputs, and w_{a_i} is the process noise.

Since w_{a_i} disturbs only the plant states, it is given by

$$w_{a_i} = \begin{bmatrix} w \\ 0 \end{bmatrix} \qquad (4.2.2)$$

where w is Gaussian with

$$w \sim N[0, W_d]. \tag{4.2.3}$$

The new controls are calculated by

$$u_{n_i} = C_i x_{a_i}, \tag{4.2.4}$$

i.e., the new control inputs are linear combinations of the plant state and the held controls.

The feedback gains, C_i, are to be selected to minimize the quadratic cost (at stochastic steady state)

$$J = E\left\{ \sum_{i=0}^{k-1} [x_{a_i}^T \; u_{n_i}^T] \begin{bmatrix} Q_{a_i} & M_{a_i} \\ M_{a_i}^T & R_{a_i} \end{bmatrix} \begin{bmatrix} x_{a_i} \\ u_{n_i} \end{bmatrix} \right\} \tag{4.2.5}$$

and, in addition, to satisfy a constraint that remains to be specified.

In order to solve this problem, substitute

$$\begin{bmatrix} x_{a_i} \\ u_{n_i} \end{bmatrix} = \begin{bmatrix} I \\ C_i \end{bmatrix} x_{a_i} \tag{4.2.6}$$

into the expression for J, to get

$$J = E\left\{ \sum_{i=0}^{k-1} x_{a_i}^T \hat{Q}_{a_i} x_{a_i} \right\} \tag{4.2.7}$$

where

$$\hat{Q}_{a_i} = [I \; C_i^T] \begin{bmatrix} Q_{a_i} & M_{a_i} \\ M_{a_i}^T & R_{a_i} \end{bmatrix} \begin{bmatrix} I \\ C_i \end{bmatrix}. \tag{4.2.8}$$

The cost can be written as

$$J = \mathrm{tr}\left[\sum_{i=0}^{k-1} \hat{Q}_{a_i} S_{a_i}\right] \qquad (4.2.9)$$

where S_{a_i} is the covariance of the augmented state

$$S_{a_i} = E\{x_{a_i} x_{a_i}^T\} \qquad (4.2.10)$$

(details of the calculation of S_{a_i} are given in chapter 6).

The only unknowns in the expressions for $\hat{Q}_{a_i}$ and S_{a_i} are the feedback gain matrices, C_i. Since the cost is a function of $\hat{Q}_{a_i}$ and S_{a_i}, its only independent variables are the feedback gains, C_i.

$$J = J(C_i) \qquad (4.2.11)$$

By grouping the C_i into an "independent parameter matrix", K, J can be written as

$$J = J(K). \qquad (4.2.12)$$

The closed form expression for this function is messy. This is primarily due to the complexity of the closed form expressions for the augmented state covariance.

Now that the cost has been expressed in terms of the feedback gains, the minimization procedure can follow the standard steepest descent algorithm.

1. Select a stabilizing K_0.

2. Evaluate $\partial J/\partial K$.
3. Calculate a new value for K according to:

$$K_{i+1} = K_i - \alpha \frac{\partial J}{\partial K} \qquad (4.2.13)$$

4. If $\|\partial J/\partial K\|$ is greater than some selected maximum value, return to step 2.
5. STOP

This method is surprisingly flexible in solving for feedback gains with arbitrary constraints. Once J is expressed in terms of the constrained gains, a minimizing value for those gains can be found.

Numerous methods exist for selecting the parameter α. The one used by MULTIRAT is a modified version of the Davies-Swann-Campey algorithm (see Adby and Dempster, 1974, pp 36-37).

4.3 The Optimal Plant-State-Only Regulator

The first restriction to the structure of the optimal multirate regulator will be to use only the plant state (and not the held controls) in the calculation of all new controls. The new controls can then be expressed as

$$u_{n_i} = C_i \begin{bmatrix} x_i \\ u_{h_i} \end{bmatrix} = [C_i' \;\; 0] \begin{bmatrix} x_i \\ u_{h_i} \end{bmatrix}. \qquad (4.3.1)$$

Since the C_i' are allowed to be periodically time varying, the parameter matrix can be written as

$$K = \begin{bmatrix} C_0' \\ C_1' \\ \cdots \\ C_{k-1}' \end{bmatrix} \qquad \text{DIM}(l \times n). \qquad (4.3.2)$$

From a mathematical point of view, there are $l*n$ independent parameters that affect J, and, for convenience, they are grouped into the "l by n" parameter matrix, K.

Since the structure of the optimal plant-state-only regulator is very similar to the structure of the unconstrained optimal regulator, the difference in performance should be relatively small. Examination of the results of the analysis of the example systems in chapters 7 and 8 shows that this is indeed true.

4.4 The Optimal Constant-Gain Regulator

Another possible simplification to the structure of the optimal multirate regulator is to force the feedback gains to remain constant over the control period.

Since the held controls can be used in the control calculations, the parameter matrix is given by

$$K = C \qquad \text{DIM}(m \times (n+m)) \qquad (4.4.1)$$

and the controls can be expressed as

$$u_{n_i} = C_i'' \begin{bmatrix} x_i \\ u_{h_i} \end{bmatrix} \qquad (4.4.2)$$

where C_i'' is obtained from C by eliminating the columns that correspond to new controls and the rows that correspond to

held controls. Note that, although the individual elements of C'' are constant, its dimensions vary from one cycle to the next. Also, any columns of C that correspond to controls that are updated on every cycle will be zero since these controls are never held and are therefore never used in the calculation of new controls.

4.5 The Optimal Constant-Gain Plant-State-Only Regulator

The final constrained regulator eliminates both of the unique features of the optimal unconstrained regulator. The feedback gains are forced to be constant and only the plant state is used to calculate the new controls.

The parameter matrix is given by

$$K = C \qquad \text{DIM}(m \times n) \qquad (4.5.1)$$

and the new controls can be expressed as

$$u_{n_i} = C_i''' x_i \qquad (4.5.2)$$

where C_i''' is obtained from C by eliminating the rows that correspond to held controls.

4.6 Discussion of Characteristics

The three optimal constrained regulators were: (in order of decreasing number of free parameters)

1. Plant-state-only - $1*n$ free parameters.
2. Constant-gain - $m*(n+m)$ free parameters.
3. Constant-gain plant-state-only regulator - $m*n$ free

parameters.

It might be expected that since the number of free parameters in the plant-state-only regulator is greater than in the constant-gain regulator, its performance will be better. Although this is usually true, it must be remembered that these regulators constrain different features of the optimal regulator and a particular regulator's performance depends on the sensitivity of the system to the specific constraint being applied.

In the case of the constant-gain plant-state-only regulator, both constraints have been applied and, as a result, the performance index for this regulator will always be greater than for the other two.

It can be seen from the examples in chapters 7 and 8 that, as expected, a regulator's performance is directly related to its structure. As the structure is simplified, the regulator's performance index increases.

4.7 Summary

In this chapter a numerical method for the solution of the optimal constrained regulator was developed. This method was then used to obtain the solution for three different constrained versions of the optimal multirate regulator.

Due to the numerical nature of the method and the complexity of many of the matrix expressions, the solution of the optimal constrained regulator can be computationally expensive. This complexity, however, only exists during the

design of the regulator. Once the optimal gains have been found, they can be easily implemented.

Although the performance of each of the constrained regulators is not as good as the optimal multirate regulator, their structures are easier to implement. The decision of which regulator is best in any given situation will depend on the level of performance required and the capabilities of the control computer.

Chapter 5

SUBOPTIMAL MULTIRATE REGULATORS

5.1 Introduction

The optimal constrained regulators considered in the previous chapter benefit from the fact that their simpler structure is easier to implement and requires less computational resources for gain scheduling and control calculations. Unfortunately, the need to use numerical algorithms in their solution means that the cost of computation in their design can become quite large. This is especially true if the designer is examining system performance for a variety of parameters (while doing sensitivity analysis, for example).

This chapter will examine three regulators that keep the benefits of those of the previous chapter (i.e. simple structure) but use alternate methods for the selection of gains. These regulators are called "suboptimal" because they no longer seek to minimize the performance index (J).

5.2 The Suboptimal Plant-State-Only Regulator

Consider once again the solution of the optimal multirate regulator as discussed in section 3.6. Solution of the optimal equivalent time invariant system resulted in a control vector given by

$$U_{m(j)} = C_m x_{(j)}. \qquad (5.2.1)$$

This equation relates the reduced control vector to the state at the beginning of the control period. That is, given the state at the beginning of the control period, all controls to be applied over the control period can be calculated.

Due to the presence of plant disturbances, better system performance can be obtained by calculating the control vectors at each cycle using the present plant state rather than the state at the beginning of the control period. The desired control vectors will be of the form

$$U_i = K_i X_i \tag{5.2.2}$$

where U_i, X_i, and K_i are referenced to the beginning of the control period.

Using the transformation matrix, E, $U_{m(j)}$ can be written in terms of the unreduced control vector U as

$$U_{(j)} = EU_{m(j)} = EC_m X_{(j)} = CX_{(j)} \tag{5.2.3}$$

where

$$C = EC_m. \tag{5.2.4}$$

While dealing with a single control period, the bracketed subscripts can be discarded.

$$U_n = CX_n \tag{5.2.5}$$

or, in expanded form

$$\begin{bmatrix} U_n \\ U_{n+1} \\ \cdots \\ U_{n+k-1} \end{bmatrix} = \begin{bmatrix} C_0 \\ C_1 \\ \cdots \\ C_{k-1} \end{bmatrix} x_n. \qquad (5.2.6)$$

This gives expressions for the control vectors at each cycle in the control period.

$$\begin{aligned} U_n &= C_0 x_n \qquad (5.2.7) \\ U_{n+1} &= C_1 x_n \\ &\vdots \\ U_{n+k-1} &= C_{k-1} x_n \end{aligned}$$

In a deterministic system x_{n+i} can be related to x_n by

$$x_{n+i} = (\phi+\Gamma K_{i-1}) x_{n+i-1} \qquad (5.2.8)$$

or

$$x_{n+i-1} = (\phi+\Gamma K_{i-1})^{-1} x_{n+i} \qquad (5.2.9)$$

then

$$\begin{aligned} x_n &= (\phi+\Gamma K_0)^{-1} x_{n+1} \qquad (5.2.10) \\ x_n &= (\phi+\Gamma K_0)^{-1}(\phi+\Gamma K_1)^{-1} x_{n+2} \\ &\vdots \\ x_n &= (\phi+\Gamma K_0)^{-1}(\phi+\Gamma K_1)^{-1} \cdots (\phi+\Gamma K_{k-2})^{-1} x_{n+k-1}. \end{aligned}$$

Replacing x_n in the equations $U_{n+i} = C_i x_n$ by the appropriate equation for x_n, the feedback gains for each cycle in the control period are obtained.

$$
\begin{aligned}
u_n &= K_0 x_n \\
u_{n+1} &= K_1 x_{n+1} \\
&\vdots \\
u_{n+k-1} &= K_{k-1} x_{n+k-1}
\end{aligned}
\tag{5.2.11}
$$

with

$$
\begin{aligned}
K_0 &= C_0 \\
K_1 &= C_1(\phi+\Gamma K_0)^{-1} \\
K_2 &= C_2(\phi+\Gamma K_0)^{-1}(\phi+\Gamma K_1)^{-1} \\
&\vdots \\
K_{k-1} &= C_{k-1}(\phi+\Gamma K_0)^{-1}(\phi+\Gamma K_1)^{-1} \ \ldots \ (\phi+\Gamma K_{k-2})^{-1}.
\end{aligned}
\tag{5.2.12}
$$

These expressions specify the feedback gains for each cycle in the control period. They can be written in compact form as

$$u_{n+i} = K_i x_{n+i} \qquad (0 \le i < k) \tag{5.2.13}$$

$$K_i = C_i \prod_{j=0}^{i-1} \{(\phi+\Gamma K_j)^{-1}\} \qquad (0 \le i < k). \tag{5.2.14}$$

Note that these equations specify both the new control inputs and the control inputs that are to be held. This appears to defeat the initial goal of specifying only the new control inputs. On closer inspection of these equations and the state equation, it can be seen that this does not present a problem. For control inputs that are to be held, these equations give a constant value over the necessary

number of cycles.

In a deterministic system, this plant-state-only regulator will provide the same performance as the optimal regulator that uses the plant state and the held controls in the calculation of the new controls. In a stochastic system, however, the restriction of using only the plant state in the calculation of the new controls results in a regulator that is suboptimal. This is due to the fact that for the plant-state-only regulator, knowledge of the held controls is implicit in the plant state but is contaminated by process noise. If the effect of this contamination is small, the two regulators have nearly identical performances. In addition, the plant-state-only regulator does have two advantages over the optimal regulator.

1. The structure of the plant-state-only regulator is simpler than that of the optimal regulator.
2. The feedback gains are easier to calculate since they do not require the propagation of the periodic Riccati equation over the control period.

5.3 The Averaged-Gains Regulator

Although the feedback gains of the optimal multirate regulator are periodically time varying, examination of the gains for the example systems of chapters 7 and 8 shows that these variations are usually relatively small. As a result, the constant gain obtained by averaging the periodic gains should provide reasonable system performance.

$$C = \frac{1}{k} \sum_{i=0}^{k-1} C_i \qquad (5.3.1)$$

where C_i are the optimal periodic feedback gains.

This regulator uses the plant state and the held controls in the calculation of the new controls and therefore has the same structure as the optimal constant-gain regulator. Due to the fact that the averaged-gains regulator no longer minimizes the performance index, it will not perform as well as the optimal constant-gain regulator.

Although this regulator is suboptimal, it is computationally less expensive to solve than the optimal constant-gain regulator and, again, its structure is simpler than the optimal multirate regulator.

5.4 The Separated-Modes Regulator

One of the possible reasons for using a multirate regulator over a single-rate regulator is the reduction in computational load due to the fact that held controls do not have to be recalculated. Depending upon the system characteristics, the multirate regulator can perform very nearly as well as the single-rate regulator. The difference in performance can, of course, be easily determined by solving the optimal single-rate regulator.

Another multirate regulator that is worth considering is the one obtained by using the gains of the optimal single-rate regulator directly. A "separated-modes" regulator will be designed by solving the optimal

single-rate regulator problem at each of the system sample rates. Each control input in the separated-modes regulator will then use the feedback gains of the single-rate regulator having the same sample rate. This is equivalent to assuming that there is no coupling between system modes so that single-rate regulators can be independently designed for each of the modes of the system.

If the assumption of no coupling between system modes is exact, this regulator should be identical to the optimal multirate regulator. It is also expected that if the coupling between modes is weak, the separated-modes regulator should give near optimal performance.

The structure of the separated-modes regulator is the same as that of the optimal constant-gain plant-state-only regulator of the previous chapter. Note that since this regulator is suboptimal and also the most tightly constrained, it is expected that it will be the poorest performer of all of the regulators considered so far.

Although the separated-modes regulator is the poorest performer, it is, by far, the easiest to solve. In addition, its structure is time invariant and uses only the plant state in new control calculations. As a result, it is also the easiest to implement.

5.5 Summary

This chapter examined three suboptimal multirate regulators. Each of the regulators considered was analagous in structure to one of the optimal constrained regulators of

the previous chapter. Although the regulators in this chapter provide suboptimal performance, they are much easier to solve than those of the previous chapter.

The first regulator considered was constrained to use the plant-state-only in control calculations and to have identical eigenvalues to the optimal multirate regulator.

The second regulator considered used the average of the optimal multirate gains. Although it required just as much work to solve as the optimal multirate regulator, its structure made it easier to implement.

The final regulator considered was one in which the optimal single-rate gains were used directly in the calculations of the multirate controls. This regulator is the poorest performer but is the easiest to solve and implement.

Chapter 6

MULTIRATE SYSTEM DESIGN AND ANALYSIS

6.1 Introduction

In order to gain a full appreciation for the characteristics of the multirate system, the techniques used in its design and analysis must be reviewed in detail. In this chapter these techniques will be discussed in a step by step manner. The procedures begin with a review of the structure of the continuous-time plant and performance index equations. Since a discrete-time regulator is to be designed, the state equation and cost functional must be converted to discrete-time form. Also, the effects of the continuous process noise must be integrated over one cycle to obtain a discrete-time representation for it. Once a discrete-time representation for the plant and the cost functional are obtained, they are converted to diagonal form and the LQG formulations are applied to obtain an optimal regulator. As part of the solution of the optimal regulator, an algebraic Riccati equation must be solved. The eigenvector method of Laub (1979) is quoted for that purpose. With the solution of the algebraic Riccati equation, a set of optimal feedback gains can be directly calculated and the solution of the optimal regulator is complete.

Once a set of feedback gains are available (whether optimal or suboptimal), the system performance can be calculated. The state covariance is first determined by solving a Lyapunov equation and is then used in the

calculation of the cost functional. The cost functional, once calculated, will be the primary tool for evaluating and comparing the performance of each of the multirate regulators.

6.2 Conversion to Discrete-time Form

Consider the discrete-time control of a continuous-time plant as depicted in Figure 6.1. The state equation for the continuous-time plant is given by

$$\dot{x}(t) = F(t)x(t) + G(t)u(t) + G_w(t)w(t) \qquad (6.2.1)$$

$$w(t) \sim N[0,W] \qquad (6.2.2)$$

where the state vector, $x(t)$, the control vector, $u(t)$, and the process noise vector, $w(t)$, belong to n, m, and r dimensional real vector spaces respectively. It is assumed that the process noise is white, with zero mean, and a Gaussian distribution. For simplicity, it is further assumed that the state transition matrix, $F(t)$, the control distribution matrix, $G(t)$, and the disturbance distribution matrix, $G_w(t)$, are time invariant.

In order to digitally control the plant, the continuous-time state, $x(t)$, is sampled at the discrete times iT, to produce the sampled state, x_i. The regulator is designed to use these samples to calculate the discrete-time controls, u_i, which are applied to the plant through zero-order hold (ZOH) circuits.

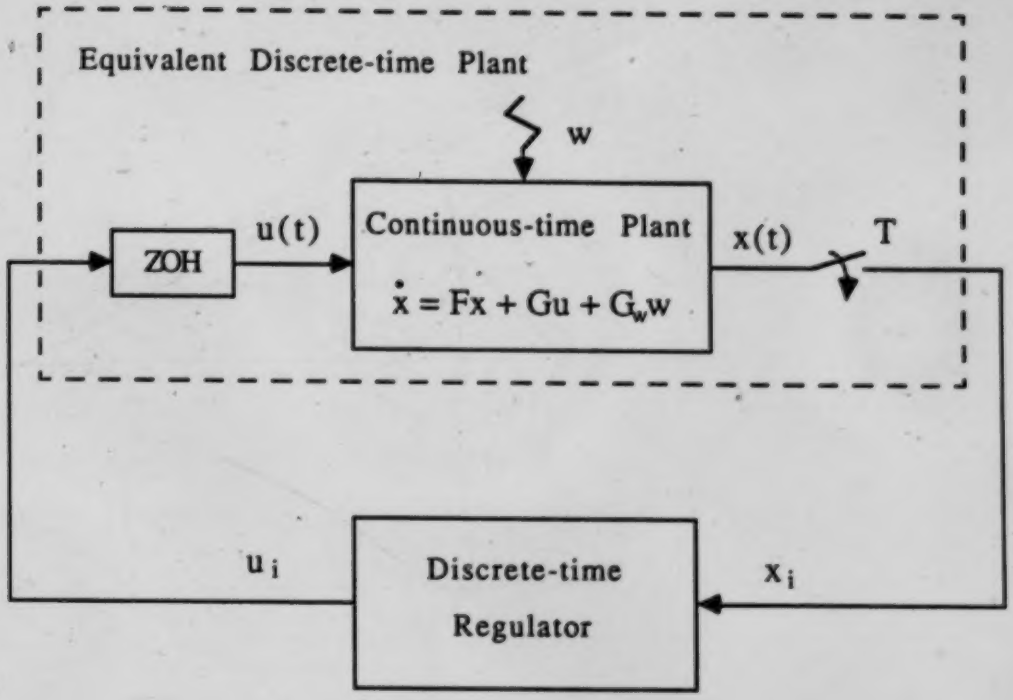

Figure 6.1 - Discrete-time Control of a Continuous-time Plant

Regulator design and system analysis is facilitated by considering the continuous-time plant, ZOH, and state sampler as a discrete-time system with inputs u_i and outputs x_i. In this system, the state is viewed to change abruptly at the sampling instants. When a state measurement is available, the regulator calculates a new control vector and supplies that vector to the zero-order hold circuits after an infinitesimal time. The ZOH circuits latch the control vector and apply it to the continuous-time plant until the plant state is sampled again and a new control vector is

available from the regulator. Once a control vector has been latched, the plant propagates in time according to its continuous-time state equation.

Solution of the differential state equation provides an expression for the propagation of the plant state between control updates.

$$x(t+\tau) = \phi(t+\tau,t)x(t) + \Gamma(t+\tau,t)u(t) + w(t,\tau) \quad (6.2.3)$$

where $w(t,\tau)$ is the integrated effect of the continuous-time process noise. (Details of the method of solution can be found in Franklin and Powell, 1980.) Due to the fact that $F(t)$ and $G(t)$ are time invariant,

$$\phi(t+\tau,t) = \exp(F\tau) \quad (6.2.4)$$

and

$$\Gamma(t+\tau,t) = \int_0^\tau \exp(F\tau)G d\tau. \quad (6.2.5)$$

The behaviour of the discrete-time plant (continuous-time plant, ZOH, and state sampler) can be modelled by a matrix difference equation.

$$x_{i+1} = \phi x_i + \Gamma u_i + w_i \quad (6.2.6)$$

If the states are sampled and the new controls applied every T seconds, the discrete-time state transition matrix is given by

$$\Phi = \exp(FT) \tag{6.2.7}$$

and the discrete-time control distribution matrix is given by

$$\Gamma = \int_0^T \exp(F\tau)G\,d\tau. \tag{6.2.8}$$

The discrete-time process noise, w_i, is the integrated effect of the continuous process noise. It can be modelled as an impulse having a Gaussian distribution with zero mean and covariance, W_d.

$$w_i \sim N[O, W_d] \tag{6.2.9}$$

where the covariance is given by

$$W_d = E\{w_i w_i^T\} = \int_0^T \exp(F\tau)G_c W G_c^T \exp(F\tau)^T d\tau. \tag{6.2.10}$$

Calculation of the integral in the expression for the control distribution matrix can be avoided by writing the state equation as, (the process noise effects are temporarily ignored),

$$\begin{bmatrix} x(t+\tau) \\ u(t+\tau) \end{bmatrix} = \Phi(\tau)\begin{bmatrix} x(t) \\ u(t) \end{bmatrix} \tag{6.2.11}$$

$$\Phi(\tau) = \exp\left[\begin{bmatrix} F & G \\ O & O \end{bmatrix}\tau\right] = \begin{bmatrix} \Phi(t+\tau,t) & \Gamma(t+\tau,t) \\ O & I \end{bmatrix}. \tag{6.2.12}$$

The equivalence of this expression to the earlier expressions for ϕ and Γ can be seen by comparing the terms of its series expansion with those of the series expansion of ϕ and Γ.

At the end of the cycle, but before the controls have been updated, the sampled value of the state will be given by

$$\begin{bmatrix} x_{i+1} \\ u_i \end{bmatrix} = \Phi(T) \begin{bmatrix} x_i \\ u_i \end{bmatrix} \tag{6.2.13}$$

with

$$\Phi(T) = \exp\left[\begin{bmatrix} F & G \\ O & O \end{bmatrix} T\right]. \tag{6.2.14}$$

Now that a discrete-time representation for the plant has been obtained, the design of the regulator can be considered. It is assumed that the regulator has perfect knowledge of the state x_i at time t_i. Using this knowledge, the regulator computes the control vectors, u_i, to be provided to the plant. In the case of an optimal regulator, the control vectors are calculated to minimize the quadratic cost

$$J = \lim_{t \to \infty} \frac{1}{t} E\left[\int_0^t (x(\tau)^T Q_A x(\tau) + u(\tau)^T R_A u(\tau)) d\tau\right] \tag{6.2.15}$$

where the continuous-time state weighting matrix, Q_A, and the continuous-time control weighting matrix, R_A, are given.

In order for J to possess a physically meaningful positive value, it is required that Q_A be positive semidefinite, and R_A be positive definite.

The cost of control can be expressed in terms of the discrete time variables by decomposing it into the sum of N integrals.

$$J = \lim_{N\to\infty} \frac{1}{NT} E\left[\sum_{i=0}^{N-1} j_i \right] \tag{6.2.16}$$

where

$$j_i = \int_{iT}^{(i+1)T} [x(\tau)^T u(\tau)^T] \begin{bmatrix} Q_A & O \\ O & R_A \end{bmatrix} \begin{bmatrix} x(\tau) \\ u(\tau) \end{bmatrix} d\tau. \tag{6.2.17}$$

Assuming that the feedback gains are stabilizing, the expected cost over any cycle will be constant. Then

$$J = \frac{1}{T} E\left[j_i \right]. \tag{6.2.18}$$

Now by expressing the state and control vectors in terms of their discrete-time counterparts,

$$j_i = \int_{iT}^{(i+1)T} [x_i^T u_i^T] \Phi(\tau)^T \begin{bmatrix} Q_A & O \\ O & R_A \end{bmatrix} \Phi(\tau) \begin{bmatrix} x_i \\ u_i \end{bmatrix} d\tau. \tag{6.2.19}$$

Since x_i and u_i are invariant over the range of integration,

$$j_i = [x_i^T \; u_i^T]\begin{bmatrix} Q & M \\ M^T & R \end{bmatrix}\begin{bmatrix} x_i \\ u_i \end{bmatrix}. \qquad (6.2.20)$$

Where the discrete-time state weighting matrix, Q, control weighting matrix, R, and cross weighting matrix, M, are given by

$$\begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} = \int_0^T \Phi(\tau)^T \begin{bmatrix} Q_A & O \\ O & R_A \end{bmatrix} \Phi(\tau) d\tau. \qquad (6.2.21)$$

Note that the integrand is sample invariant so that Q, M, and R are constant matrices and the integration can occur over the interval $[O,T]$.

The integrals specifying the discrete process noise covariance and discrete weighting matrices could be solved by one of the popular numerical methods (eg. Simpson or Runga-Kutta) but a cleaner and more accurate method will be used here. This method of computing integrals involving matrix exponentials was described by Van Loan (1978). Using Van Loan's notation, the method is described as follows.

INTEGRALS INVOLVING THE MATRIX EXPONENTIAL

Let A and Q_c be real matrices of dimension $n \times n$. Assume that Q_c is symmetric ($Q_c^T = Q_c$) and positive semidefinite ($x^T Q_c x \geq 0$). The integral

$$Q(\Delta) = \int_0^\Delta \exp(A^T s) Q_c \exp(As)\, ds \tag{6.2.22}$$

can be obtained by calculating the exponential

$$\exp\left[\begin{bmatrix} -A^T & Q_c \\ O & A \end{bmatrix}\Delta\right] = \begin{bmatrix} F_2(\Delta) & G_2(\Delta) \\ O & F_3(\Delta) \end{bmatrix}. \tag{6.2.23}$$

It can be shown that

$$Q(\Delta) = F_3(\Delta)^T G_2(\Delta). \tag{6.2.24}$$

The expression for W_d can be cast into this form by writing

$$W_d = \int_0^T \exp(F\tau) G_c W G_c^T \exp(F\tau)^T d\tau \tag{6.2.25}$$

and making the following variable associations.

$$A \leftarrow F^T$$
$$Q_c \leftarrow G_v W G_v^T \qquad (6.2.26)$$
$$\Delta \leftarrow T.$$

Then

$$W_d = F_3(T)^T G_2(T). \qquad (6.2.27)$$

The discrete-time weighting matrices are obtained by writing

$$\begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} = \int_0^T \Phi(\tau)^T \begin{bmatrix} Q_A & O \\ O & R_A \end{bmatrix} \Phi(\tau) d\tau \qquad (6.2.28)$$

$$\Phi(\tau) = \exp\left[\begin{bmatrix} F & G \\ O & O \end{bmatrix} \tau\right] \qquad (6.2.29)$$

with the variable associations

$$A \leftarrow \begin{bmatrix} F & G \\ O & O \end{bmatrix}$$
$$Q_c \leftarrow \begin{bmatrix} Q_A & O \\ O & R_A \end{bmatrix} \qquad (6.2.30)$$
$$\Delta \leftarrow T.$$

Then

$$\begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} = F_3(T)^T G_2(T). \qquad (6.2.31)$$

6.3 The Discrete-time Riccati Equation

After conversion to discrete time, the state equation and cost functional can be written as

$$x_{i+1} = \phi_i x_i + \Gamma_i u_i + w_i \tag{6.3.1}$$

$$J = \frac{1}{T} E\left\{ [x_i^T \; u_i^T] \begin{bmatrix} Q_i & M_i \\ M_i^T & R_i \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix} \right\}. \tag{6.3.2}$$

The problem of designing a regulator to minimize J is called the "Linear Quadratic Gaussian (LQG) Regulator Problem". Its solution is well understood and is described in detail in Levis and Dorato (1971). The optimal control vector, u_i, is given by

$$u_i = -(R_i + \Gamma_i^T K_{i+1} \Gamma_i)^{-1} (\Gamma_i^T K_{i+1} \phi_i + M_i^T) x_i = C_i x_i \tag{6.3.3}$$

and the difference equation satisfied by K_i is called the "discrete Riccati equation"

$$\begin{aligned} K_i = {} & (\phi_i^T K_{i+1} \phi_i + Q_i) - (\Gamma_i^T K_{i+1} \phi_i + M_i^T)^T \\ & \cdot (R_i + \Gamma_i^T K_{i+1} \Gamma_i)^{-1} (\Gamma_i^T K_{i+1} \phi_i + M_i^T) \qquad 0 < i < N. \end{aligned} \tag{6.3.4}$$

Of special interest here is the optimal regulator problem for time invariant or periodically time varying systems with an infinite optimization interval. If the system is controllable, † the discrete Riccati equation will possess time invariant and periodically time varying

---

† For a detailed discussion of system controllability and its implications, see Kailath (1980).

solutions respectively, and the control laws, as well, will be time invariant and periodically time varying.

The discrete Riccati equation can be solved recursively in the backward direction by starting with an initial estimate for K_N. For the case of an infinite optimization interval,

$$K_N = O. \tag{6.3.5}$$

The Riccati equation is then propagated until a steady state solution is obtained. Note that this method is valid for both time varying and time invariant systems but it is a numerical solution and can require a large amount of computation if an accurate steady state value is sought. In the next section, a different method for obtaining the steady state solution for a time invariant system will be given.

In the case where all matrices defining the optimal regulator problem are time invariant, the steady state solution of the discrete Riccati equation will also be time invariant. It can be written as

$$K = (\phi^T K \phi + Q) - (\Gamma^T K \phi + M^T)^T (R + \Gamma^T K \Gamma)^{-1} (\Gamma^T K \phi + M^T). \tag{6.3.6}$$

In this form, it is called the "algebraic Riccati equation". The time invariant control law is then given by

$$u_i = -(R + \Gamma^T K \Gamma)^{-1} (\Gamma^T K \phi + M^T) x_i = C x_i. \tag{6.3.7}$$

6.4 Solution of the Algebraic Riccati Equation

The method of solution of the algebraic Riccati equation used here is attributed to Laub (1979). It is similar to the classical eigenvector method described in Franklin and Powell (1980), but is numerically more attractive due to its use of Schur vectors rather than eigenvectors.

The algebraic Riccati equation to be solved is

$$\hat{K} = (\phi^T \hat{K} \phi + Q) - (\Gamma^T \hat{K} \phi)^T (R + \Gamma^T \hat{K} \Gamma)^{-1} (\Gamma^T \hat{K} \phi). \qquad (6.4.1)$$

In outline form, the solution proceeds as follows.

1. Write the system control Hamiltonian as

$$H_c = \begin{bmatrix} \phi + \Gamma R^{-1} \Gamma^T \phi^{-T} Q & -\Gamma R^{-1} \Gamma^T \phi^{-T} \\ -\phi^{-T} Q & \phi^{-T} \end{bmatrix}. \qquad (6.4.2)$$

2. Convert H_c to real Schur form

$$U^T H_c U = S = \begin{bmatrix} S_{11} & S_{12} \\ O & S_{22} \end{bmatrix}. \qquad (6.4.3)$$

where the spectrum of S_{11} lies inside the unit circle and the spectrum of S_{22} lies outside the unit circle and where the matrix of Schur vectors is given by

$$U = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix}. \qquad (6.4.4)$$

3. The symmetric, positive definite solution of the discrete algebraic Riccati equation is then given by

$$\bar{K} = U_{21}U_{11}^{-1}. \qquad (6.4.5)$$

The classical eigenvector method is analogous with H_c being converted to diagonal form rather than real Schur form. In that case, of course, U contains the eigenvectors of H_c rather than the Schur vectors.

The Riccati equation solved here is a special case of the Riccati equation discussed in the previous section. They are identical if $M=O$ or, in other words, if the cost functional weighting matrix is block diagonal. Unfortunately, M appeared in the conversion to discrete-time form and will generally be nonzero if the cost functional is originally specified in terms of the continuous-time system variables. This limitation of the eigenvector methods can be overcome by transforming the system state and control variables to obtain an equivalent system having a block diagonal weighting matrix.

6.5 Conversion to Diagonal Form

A method for transforming the optimal regulator problem into an equivalent problem having a block diagonal weighting matrix in the cost functional is described in Kwakernaak and Sivan (1972) and in Katz (1974). The method is repeated and expanded in this section.

Given the state equation

$$x_{i+1} = \phi x_i + \Gamma u_i \qquad (6.5.1)$$

and the cost functional

$$J = \frac{1}{T} E\left\{ \begin{bmatrix} x_i^T & u_i^T \end{bmatrix} \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix} \right\}, \tag{6.5.2}$$

an equivalent system can be written as

$$x_{i+1} = (\phi - \Gamma R^{-1} M) x_i + \Gamma u_i \tag{6.5.3}$$

with

$$J = \frac{1}{T} E\left\{ \begin{bmatrix} x_i^T & u_i^T \end{bmatrix} \begin{bmatrix} \Omega & O \\ O & R \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix} \right\} \tag{6.5.4}$$

where

$$u_i = u_i + R^{-1} M^T x_i \tag{6.5.5}$$

and

$$\Omega = Q - M R^{-1} M^T. \tag{6.5.6}$$

If the regulator for this diagonalized system is written as

$$u_i = c x_i, \tag{6.5.7}$$

the regulator for the original system will be given by

$$u_i = (c - R^{-1} M^T) x_i. \tag{6.5.8}$$

The optimal discrete-time regulator is now solved by converting the original problem to its equivalent block diagonal form. Next, the Riccati equation for the equivalent system is solved by using the Schur vector method of the previous section. The feedback gains for the original system can then by obtained by adding the term $R^{-1}M^T$ to the feedback gains calculated for the diagonal system.

Another interesting feature of the equivalent diagonal system is that its Riccati equation is identical to the Riccati equation of the original system. This can be seen by writing the Riccati equation of the equivalent system, in terms of the original system matrices. With a bit of algebraic manipulation, the Riccati equations of the original system and the equivalent system can be shown to be identical. As a result, the controls can also be expressed as

$$u_i = -(R + \Gamma^T K \Gamma)^{-1}(\Gamma^T K \phi + M^T)x_i \qquad (6.5.9)$$

where K is the solution of the algebraic Riccati equation of the diagonalized system. This fact is especially important in the solution of the optimal multirate regulator. In that problem, the time varying multirate system is first converted to an equivalent time invariant form. The Riccati equation for the equivalent form is then solved by transforming it to block diagonal form and then using the Schur vector method of solution. At this point, it is the Riccati equation solution that is of interest and not the feedback gains since the Riccati equation must be propagated

over one control period in order to obtain the feedback gains at each cycle.

6.6 System Response to Process Noise

By using the optimal discrete-time regulator

$$u_i = Cx_i, \tag{6.6.1}$$

the closed-loop response of the discrete-time system will be given by

$$x_{i+1} = \phi x_i + \Gamma C x_i + w_i = (\phi + \Gamma C)x_i + w_i. \tag{6.6.2}$$

Define Φ to be the "closed-loop state transition matrix", so that

$$x_{i+1} = \Phi x_i + w_i. \tag{6.6.3}$$

If Φ is a stable matrix, i.e. all eigenvalues of Φ are within the unit circle, an expression for the state covariance can be obtained. (See also Bryson and Ho, 1975.) Define the covariance (or mean square value) of x_i to be

$$E\{x_i x_i^T\} = X_i. \tag{6.6.4}$$

Then X_i can be solved as follows.

$$E\{x_{i+1}x_{i+1}^T\} = E\{(\Phi x_i + w_i)(\Phi x_i + w_i)^T\} \tag{6.6.5}$$

It is assumed that the process noise is independent of the present state

$$E\{x_i w_i^T\} = 0 \qquad (6.6.6)$$

so that

$$E\{x_{i+1}x_{i+1}^T\} = \Phi E\{x_i x_i^T\}\Phi^T + E\{w_i w_i^T\}. \qquad (6.6.7)$$

Using the definition of the discrete process noise covariance,

$$E\{w_i w_i^T\} = W_d \qquad (6.6.8)$$

gives

$$X_{i+1} = \Phi X_i \Phi^T + W_d. \qquad (6.6.9)$$

At steady state X_i will be constant and the state covariance equation can be written as

$$X = \Phi X \Phi^T + W_d. \qquad (6.6.10)$$

This equation is linear in the elements of X and has a variety of solution methods proposed in the literature. The method used here is from Kailath (1980). It can be shown that the solution of the covariance equation is given by

$$\text{vec}(X) = (I - \Phi\otimes\Phi)\,\text{vec}(W_d) \qquad (6.6.11)$$

where vec(Δ) is obtained by "stacking" the rows of Δ and where $\Phi\otimes\Phi$ represents the Kroenecker product of Φ with itself. †

---

† For a review of the theory of Kroenecker products, see Brewer (1978).

This method is easy to implement and gives satisfactory performance for systems of small order. For systems of higher order (say, $n>5$), many of the other methods of solution are numerically superior and computationally faster.

6.7 Calculation of the Cost Functional

At steady state, the process noise will continuously disturb the plant states directly and the plant controls indirectly. As a result, the cost functional will never go to zero but will reach a constant average value. This value can be calculated once the covariance equation has been solved (See Bryson and Ho, 1975).

The cost functional was found to be

$$J = \frac{1}{T} E\left[[x_i^T \; u_i^T] \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} \begin{bmatrix} x_i \\ u_i \end{bmatrix} \right]. \qquad (6.7.1)$$

If $u_i = Cx_i$, this can be written as

$$J = \frac{1}{T} E\{x_i^T \hat{Q} x_i\} \qquad (6.7.2)$$

where

$$\hat{Q} = [I \; C^T] \begin{bmatrix} Q & M \\ M^T & R \end{bmatrix} \begin{bmatrix} I \\ C \end{bmatrix}. \qquad (6.7.3)$$

The expected value of the quadratic form of a random variable having a Gaussian distribution can be written as

$$E\{x^T A x\} = \bar{x}^T A \bar{x} + \mathrm{tr}\{A\,\mathrm{cov}(x)\}. \tag{6.7.4}$$

Since the process noise is Gaussian with zero mean,

$$\bar{x} = 0 \tag{6.7.5}$$

and the cost functional can be written as

$$J = \frac{1}{T}\,\mathrm{tr}\{\hat{Q}X\}. \tag{6.7.6}$$

Since $1/T$ is a constant scale factor, the cost functional to be minimized can be written in the simpler form

$$J' = \mathrm{tr}\{\hat{Q}X\}. \tag{6.7.7}$$

This equation applies to both optimal and suboptimal regulators, and it proves to be very valuable in comparing the performance of regulators designed using different criteria. In contrast to many of the other possible measures of system performance, it provides a scalar measure that is easy to interpret. Naturally, the optimal regulator, as obtained by solution of the algebraic Riccati equation, should provide a minimum value for J. Regulators designed using other methods can be compared to the optimal regulator directly by solving the state covariance equation and then calculating their suboptimal value of J.

6.8 Summary

In this chapter, many of the mathematical tools needed in the design and analysis of a discrete-time regulator for a continuous-time system were given. In addition, references were made to more detailed analyses of some of the problems. The chapter began by describing the discrete-time equivalent of a continuous-time system. The linear-quadratic-Gaussian regulator problem was then stated and the continuous-time cost functional was converted to discrete-time form. At this point, the discrete Riccati equation was introduced to describe the solution of the LQG regulator problem. Two eigenvector methods for the solution of the algebraic Riccati equation were reviewed and the system description was then cast into the diagonal form to which these solutions could be applied. After the regulator was solved, its performance was calculated by solving the covariance equation and by calculating the expected value of the cost functional.

Chapter 7

EXAMPLE #1 - THE INVERTED PENDULUM

7.1 Introduction

In the previous chapters a variety of multirate regulators were studied. The difficulty of solution of each regulator was discussed and, from an examination of their structure, they were listed in order of decreasing performance. Although examination of the regulator structures allows them to be ranked from best performer to poorest performer, the amount of difference between each regulator will depend on the parameters of the system. In order to get a feel for these differences, a typical example system must be analyzed and its regulator performance quantified.

In this chapter each of the multirate regulators proposed in the previous chapters are solved for an example system. An upper bound on system performance is determined by solving the optimal single-rate regulator for the system with all controls updated at the fast rate. Likewise, a lower bound is determined by solving the optimal single-rate regulator for the system with all controls updated at the slow rate. By comparing these absolute values of performance to the performance of each of the multirate regulators, conclusions are drawn about the benefits of using a multirate regulator and about which multirate regulator provides the best cost/performance ratio.

Since it is expected that the performance of the regulator will depend both on the type of system and on the system parameters, two example systems are analyzed. In this chapter, a four state plant that has two unstable modes is considered. In the next chapter, the example plant has two undamped oscillatory modes. The system parameters are varied over a wide range to determine their effect on regulator performance. For the unstable plant of this chapter, the parameters of primary interest are:

- degree of instability,
- control effectiveness,
- ratio of state to control cost weighting,
- base sampling rate, and
- sample rate ratio.

7.2 The Inverted Pendulum

Consider the inverted pendulum of Figure 7.1. The mechanism consists of a cart upon which a small mass at the end of a weightless rod is to be balanced. Motion is allowed in the plane of the paper only and the control objective is to keep the rod in a position as close to vertical as possible. Note that the plant (consisting of cart, rod and small mass) is unstable and the small mass will fall unless stabilizing forces are applied.

This problem is popular in the literature, and both Elgerd (1967) and Fortmann (1977) provide some analyses of its characteristics. The usual method of controlling the plant is to apply a horizontal force, F, to the cart. In

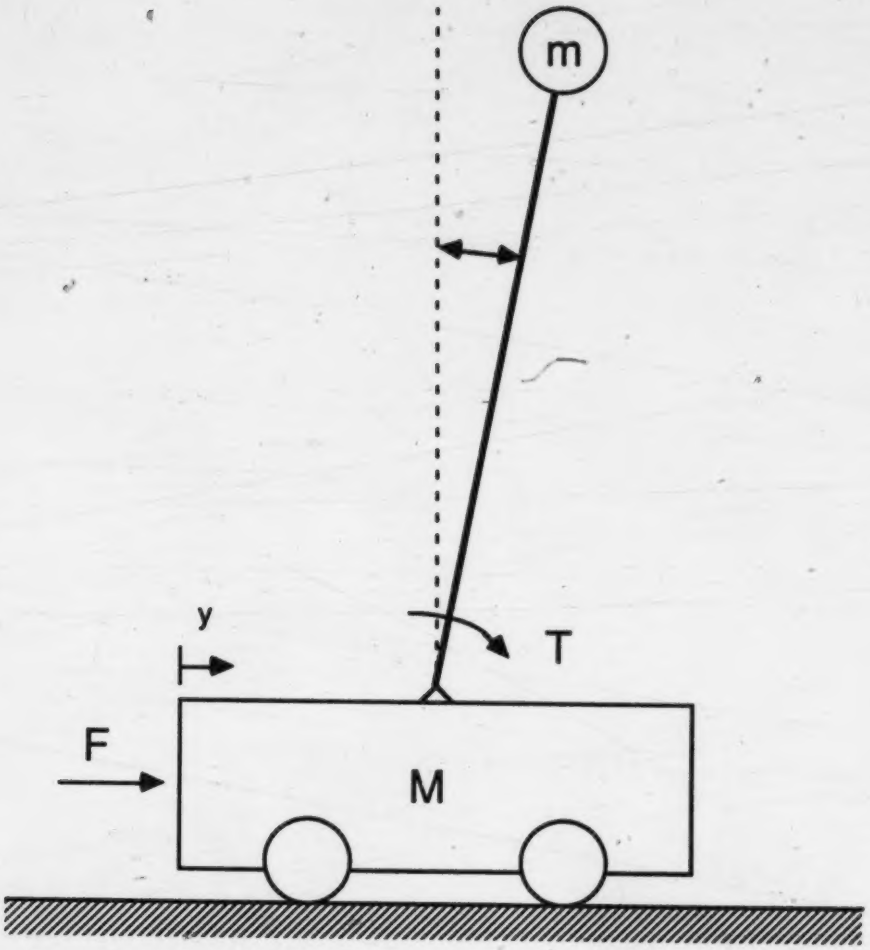

Figure 7.1 - The Inverted Pendulum

this form, the problem is identical to that of vertically balancing a broom in ones hand and is often called the "broom balancing problem". The weightless rod and small mass are then referred to as the "broom".

The problem also provides an accurate dynamic model of a space booster during take off. In that problem, the booster is stabilized by directing the rocket engine thrust to provide a small horizontal component.

To illustrate the use of multirate control, the problem will be modified by adding a small motor to the base of the broom. The motor will assist in the stabilization of the broom by applying a small torque, T. In addition to assuming that the motor is small, it will also be assumed that it is relatively fast in comparison to the motor providing the horizontal force, F, to the cart. In this way, the unstable broom can be partially stabilized by the fast control torque, T, and partially stabilized by the slower control force, F.

To approximate a real world situation, the broom will also be driven by an unpredictable wind disturbance, w. By including a disturbance, the regulator will be forced to continually provide control inputs to counteract the wind and to maintain stability.

The equations of motion for the plant are obtained by application of Lagrange's equations.

$$\frac{d}{dt}\left[\frac{\partial T}{\partial \dot{q}_i}\right] - \frac{\partial T}{\partial q_i} + \frac{\partial V}{\partial q_i} = Q_i \qquad (7.2.1)$$

The system has two degrees of freedom. The generalized coordinates are chosen as:

y - the horizontal position of the cart.

θ - the angle between the broom and vertical.

The kinetic energy of the plant is the sum of the kinetic energies of the broom and the cart.

$$T = \tfrac{1}{2}M\dot{y}^2 + \tfrac{1}{2}mv_m^2 \qquad (7.2.2)$$

The velocity of the small mass at the end of the pendulum, v_m, can be found by summing the vector velocity of the cart and the vector velocity of the small mass relative to the cart. Use of the cosine law gives

$$v_m^2 = \dot{y}^2 + l^2\dot{\theta}^2 + 2l\cos\theta\dot{\theta}\dot{y} \qquad (7.2.3)$$

and the kinetic energy of the plant is then

$$T = \tfrac{1}{2}(M+m)\dot{y}^2 + \tfrac{1}{2}ml^2\dot{\theta}^2 + ml\cos\theta\dot{\theta}\dot{y}. \qquad (7.2.4)$$

The effect of gravity on the pendulum is taken into account by the potential energy.

$$V = mgl\cos\theta \qquad (7.2.5)$$

The generalized forces, Q_i, are obtained by determining the amount of work performed during a small displacement of the generalized coordinates. For the y coordinate

$$Q_y\delta y = -F\delta y + w\delta y \qquad (7.2.6)$$

$$Q_y = F + w \tag{7.2.7}$$

and for the θ coordinate

$$Q_\theta \delta\theta = T\delta\theta + l\cos\theta w\delta\theta \tag{7.2.8}$$

$$Q_\theta = T + l\cos\theta w. \tag{7.2.9}$$

These relationships can be plugged into the Lagrange equations to obtain the equations of motion. For the y coordinate

$$(M+m)\ddot{y} + ml\cos\theta\ddot{\theta} - ml\sin\theta\dot{\theta}^2 = F + w \tag{7.2.10}$$

and for the θ coordinate

$$m\cos\theta\ddot{y} + ml\ddot{\theta} - mg\sin\theta = T/l + w\cos\theta. \tag{7.2.11}$$

It is assumed that the regulator will be able to keep the broom very close to vertical so that θ will be small. In that case, the equations can be linearized by the approximations

$$\cos\theta \approx 1 \qquad \sin\theta \approx \theta \tag{7.2.12}$$

and by eliminating terms that are greater than first order in θ or y. After linearization, the equations of motion are

$$(M+m)\ddot{y} + ml\ddot{\theta} = F + w \tag{7.2.13}$$

$$m\ddot{y} + ml\ddot{\theta} - mg\theta = T/l + w. \qquad (7.2.14)$$

Eliminating $\ddot{\theta}$ yields the cart acceleration

$$\ddot{y} = \frac{-mg}{M}\theta + \frac{1}{M}F - \frac{1}{Ml}T \qquad (7.2.15)$$

while eliminating $\ddot{y}$ yields the angular acceleration

$$\ddot{\theta} = \frac{(M+m)g}{Ml}\theta - \frac{1}{Ml}F + \frac{(M+m)}{mMl^2}T + \frac{(M+m)}{mMl}w. \qquad (7.2.16)$$

The linearized state space representation is then

$$\dot{x} = \begin{bmatrix} \dot{y} \\ \ddot{y} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & b & 0 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ c & d \\ 0 & 0 \\ e & f \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ g \end{bmatrix} w \qquad (7.2.17)$$

or

$$\dot{x}(t) = Fx(t) + Gu(t) + G_w w(t) \qquad (7.2.18)$$

where the controls have been normalized to

$$U_1 = \frac{T}{m} \qquad U_2 = \frac{F}{(m+M)} \qquad w = \frac{w}{m} \qquad (7.2.19)$$

and where

$$a = -rg \qquad b = \frac{(1+r)g}{l}$$

$$c = \frac{-r}{l} \qquad d = (1+r) \qquad (7.2.20)$$

$$e = \frac{(1+r)}{l^2} \qquad f = \frac{-(1+r)}{l}$$

$$g = \frac{(1+r)}{l} \qquad r = \frac{m}{M}.$$

The open-loop poles of the system are determined by setting the open-loop characteristic equation to zero.

$$\det(\lambda I - F) = 0. \qquad (7.2.21)$$

This gives the four open-loop poles

$$\lambda = 0, 0, \pm b^{1/2} \qquad (7.2.22)$$

where the two non-oscillating poles at $\lambda=0$ correspond to the motion of the cart and the one unstable pole and one stable pole at $\lambda=\pm b^{1/2}$ correspond to the motion of the pendulum. (This can be seem by considering the motion of the center of mass of the system.)

It is also interesting to take note of the controllability of the system. Intuitively, it seems that the system should be controllable through the horizontal force F. This is to be expected since, with a little practice, a person can usually balance a broom in his hand. The torque T, however, is able to control the angle θ but has no way of moving the system center of mass. For this reason, the system is not controllable through the torque

input. These intuitive expectations are easily confirmed by determining the rank of the controllability matrix.

$$C = [\, b \mid Fb \mid F^2b \mid F^3b \,]. \qquad (7.2.23)$$

When b is the column of the control distribution matrix, G, corresponding to the horizontol force, F, the matrix has full rank, but when b is the column of G corresponding to the torque, T, the matrix is singular. For these reasons, the torque input can help stabilize the broom but cannot replace the horizontal force in stabilizing the system.

7.3 A Typical Inverted Pendulum Regulator

Now that a linear state space representation for the inverted pendulum has been obtained, its multirate regulators can be designed and analyzed. To assist with this task, a software package, called "MULTIRAT", was designed and coded to take the parameters of a system and perform the following steps: †

1. Generate the system matrices from the system parameters.
2. Convert the state equation and cost functional to discrete-time form.
3. Calculate continuous and discrete-time open-loop system eigenvalues.
4. Calculate the feedback gains, closed-loop eigenvalues, and expected performance for each of the following regulators.

---

†A further description of MULTIRAT and its listing can be found in Appendix A.

a. fast rate regulator.
b. slow rate regulator.
c. optimal multirate regulator.
d. optimal plant-state-only regulator.
e. optimal constant-gain regulator.
f. optimal constant-gain plant-state-only regulator.
g. plant-state-only regulator.
h. averaged-gains regulator.
i. separated-modes regulator.

The system matrices are generated from the system parameters in a replaceable subroutine called "MODGEN". † All information needed by MULTIRAT is read from an input file and then checked and formatted by MODGEN. For the inverted pendulum, MODGEN requires the following parameters.

1. length - length of pendulum.
2. *r* - ratio of the mass of the pendulum to the mass of the cart.
3. weirat - ratio of the position state weighting to the control input weighting. For these examples both position states are weighted equally and both control inputs are weighted equally. The velocity states are given zero weighting.
4. *TT* - control period.
5. *kk* - number of cycles in one control period.
6. *SA* - a matrix describing the sampling scheme.

---

† MODGEN listings for both the inverted pendulum and oscillating carts can be found in Appendix B.

To analyze a particular multirate system, these parameters are placed in an input file and the name of that file is passed to MULTIRAT when it is run. The input file for a typical inverted pendulum can be found in Appendix C. This pendulum has the following characteristics.

1. a pendulum length of 1 meter.
2. a ratio of pendulum mass to cart mass of 0.1.
3. a ratio of state weighting to control weighting of 1000.
4. a torque update rate of 20 hertz.
5. a horizontal force update rate of 5 hertz.

MULTIRAT was run to solve each of the multirate regulators for this system. The output can be found in Appendix D.

To illustrate the difference in structure between each of the multirate regulators, their feedback gains for the fast control input have been listed in Table 7.1. Several characteristics are obvious.

1. The only regulators to use feedback gains that vary over the control period are the optimal multirate, optimal plant-state-only, and suboptimal plant-state-only regulators. All other regulators use constant feedback gains.
2. The optimal multirate, optimal constant-gain, and averaged-gains regulators are the only ones to use the held control state in the calculation of new controls.
3. The feedback gains do not vary greatly from one regulator to the next. The optimal multirate regulator, however, does reduce the gains on the y, and $\dot{y}$ states

Table 7.1 - Feedback Gains for a Typical Inverted Pendulum System

Regulator	y	$\dot{y}$	θ	$\dot{\theta}$	u_h
optimal multirate					
cycle 0	-11.87	-5.48	-33.62	-7.74	0.0
cycle 1	-3.90	-2.60	-35.71	-7.95	0.54
cycle 2	-6.21	-3.58	-35.38	-7.93	0.33
cycle 3	-8.91	-4.57	-34.72	-7.86	0.15
optimal pso					
cycle 0	-11.96	-5.49	-33.58	-7.74	0.0
cycle 1	-16.63	-6.06	-31.59	-7.96	0.0
cycle 2	-16.70	-5.81	-31.79	-8.02	0.0
cycle 3	-13.78	-5.27	-33.19	-7.93	0.0
plant-state-only					
cycle 0	-11.87	-5.48	-33.62	-7.74	0.0
cycle 1	-13.16	-5.65	-32.59	-7.62	0.0
cycle 2	-14.08	-5.72	-31.65	-7.52	0.0
cycle 3	-14.14	-5.67	-31.39	-7.52	0.0
optimal cg	-13.21	-5.40	-32.96	-7.90	0.09
optimal cgpso	-15.14	-5.76	-32.35	-7.92	0.0
averaged-gains	-7.72	-4.06	-34.86	-7.87	0.25
separated-modes	-13.23	-5.68	-32.65	-7.61	0.0
fast rate	-13.23	-5.68	-32.65	-7.61	0.0
slow rate	-9.39	-4.49	-20.94	-5.72	0.0

when it uses the held control state.

4. The optimal constant-gain regulator has access to the hold state but assigns only a small feedback gain to it. This gain is somewhat less than that used by the optimal multirate regulator and, from this, it seems that the hold state might be of somewhat less value if the feedback gains are forced to be constant.

The quality of each of these regulators can be compared by driving the pendulum with a wind disturbance and checking the expected value of the performance index. (The wind disturbance used by MULTIRAT has a power spectral density of unity.) The performance of each regulator is listed in decreasing order in Table 7.2. Again, several observations can be made.

1. For all practical purposes, each of the multirate regulators performs as well as the optimal multirate regulator. In most real world applications, a difference in performance of less than 10% is insignificant. For the inverted pendulum, even a difference of 50% would be difficult to detect visually from the motion of the pendulum. The performance of the separated-modes regulator (the poorest of the multirate regulators) is an indetectable 0.2% worse than the optimal multirate regulator. From this it seems that the hold state and time varying gains are not needed to provide near optimal control.
2. Although the differences are small, the relative

Table 7.2 - Multirate Regulator Performance for a Typical Inverted Pendulum System

Regulator	Expected Performance ($\times 10^{-5}$)	% difference from optimal multirate
fast rate	643.76	-2.6
optimal multirate	660.99	0.0
optimal pso	661.10	0.02
optimal cg	661.15	0.02
optimal cgpso	661.18	0.03
averaged-gains	661.37	0.06
plant-state-only	662.16	0.18
separated-modes	662.32	0.20
slow rate	1508.60	128.0

performance of each of the regulators is as expected. Remember that as a regulator's structure is simplified, its performance index is expected to increase. Since the optimal constant-gain plant-state-only regulator has the simplest structure of the optimal constrained regulators, its performance will be poorer than that of the optimal constant-gain or optimal plant-state-only regulators. Each suboptimal regulator performs more poorly than the optimal regulator having the same structure and, in this example, even more poorly than

the optimal regulators that have simpler structures. The suboptimal regulators are easy to solve but (from a mathematical point of view) are not quite as good as their optimal counterparts. Finally, the separated-modes regulator, while easiest to solve, provides the poorest performance.

3. A regulator that updates both inputs at the fast rate performs only 2.6% better than the optimal multirate regulator. In other words, there is little to be gained by increasing the update rate of the control force acting on the cart.
4. The slow rate regulator provides much poorer system performance than the optimal multirate regulator. Since the only difference between these regulators is their update rate of the torque input, it can be concluded that the torque input update rate cannot be reduced without adversely affecting system performance. This is to be expected since the wind disturbance effects on the pendulum will grow rapidly unless they are continually checked.

The characteristics of this example system support many of the underlying ideas of this thesis. For example, it seems that:

1. A multirate regulator can perform very nearly as well as a fast single-rate regulator for systems in which the fast control input is coupled to the fast (or unstable) mode of the system.

2. A multirate regulator having a simpler structure than the optimal multirate regulator will be adequate in many situations.

These claims, of course, must be supported by examining a wider variety of systems. In the next couple of sections, the multirate regulator for the inverted pendulum will be considered as some of the parameters of the pendulum are varied over a wide range.

7.4 Length Effects

It was noted in the last section that the update rate for the torque input had a significant effect on system performance. This is due to the fact that the pendulum is unstable and disturbance inputs will grow rapidly if they are not monitored closely enough. The rate at which these disturbances can grow depends on the degree of instability of the system, that is, the location of the pole in the right-half-plane. The pole location was found to be $\lambda = ((1+r)g/l)^{1/2}$. Thus, as l is increased, the disturbance inputs will take longer to grow and the multirate regulator should have an easier job.

The performance of the multirate regulators for pendulum lengths from 2 centimeters to 20 meters are listed in Table 7.3 and the performance of the regulators is graphed in Figure 7.2.

From the table and graph, the following observations can be made.

Table 7.3 Regulator Performance as a Function of Length

(Expected Performance × 10^{-4})

LENGTH	FAST	SLOW	OPTIMAL	OPTIMAL PSO	OPTIMAL CG	OPTIMAL CGPSO	PSO	AVERAGE	SEP MODES
0.02	122	416,000	122	122	122	122	513	122	122
0.05	16.3	6620	16.3	16.3	16.3	16.4	671	16.4	16.4
0.10	4.90	646	4.95	4.95	4.95	4.95	8.16	4.95	4.95
0.20	2.98	99.9	3.04	3.04	3.04	3.04	3.06	3.04	3.05
0.50	3.72	19.6	3.82	3.82	3.82	3.82	3.84	3.82	3.84
1.00	6.44	15.1	6.61	6.61	6.61	6.61	6.62	6.61	6.62
2.00	13.7	21.8	13.9	13.9	13.9	13.9	13.9	13.9	13.9
5.00	45.1	57.4	45.5	45.5	45.5	45.5	45.5	45.5	45.5
10.0	120	141	121	121	121	121	121	121	121
20.0	329	368	332	332	332	332	332	332	332

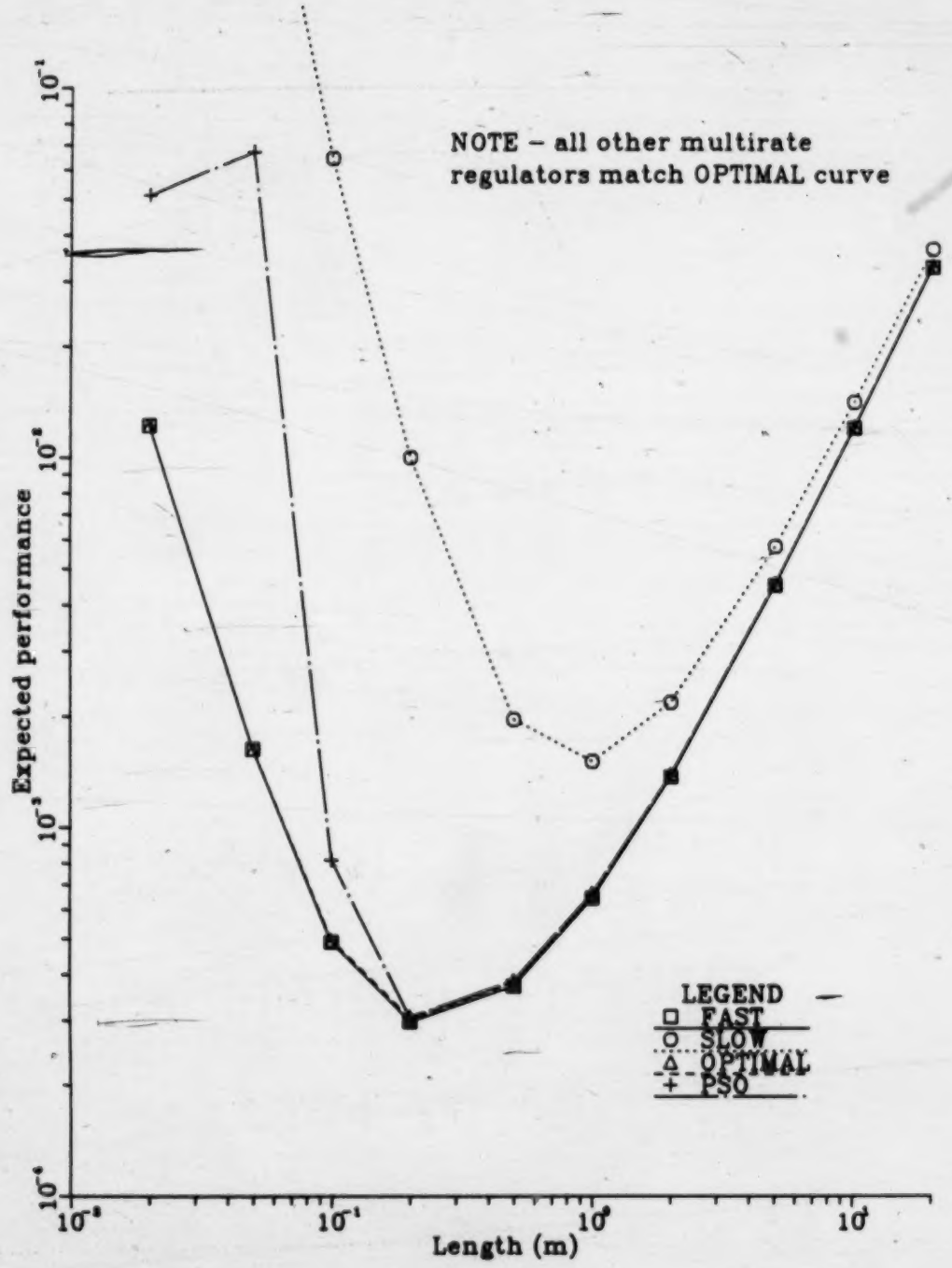

Figure 7.2 - Regulator Performance as a Function of Length

1. Even over a wide variation in length, each of the multirate regulators provides performance that is very nearly identical to that of the optimal multirate regulator. This performance is only a few percent poorer than the performance of the fast rate regulator.
2. As the pendulum length increases between 2 and 50 centimeters, the regulator performance improves by close to two orders of magnitude. This is due to a decrease in the instability of the pendulum as its length increases.
3. At lengths greater than 50 centimeters, the regulator performance rapidly gets worse. The cause of this can be seen by examining the control distribution matrix. As l is increased, the terms e and f cause a reduction in the effectiveness of the control inputs. In fact, the torque input loses effectiveness as the square of the pendulum length.
4. The difference in performance between the slow and fast rate regulators decreases as l is increased. This is again due to the fact that a higher sample rate is necessary when the system is highly unstable.
5. The plant-state-only regulator performs poorly when the pendulum is short. This also is an artifact of the increased instability of the system. Remember that the plant-state-only regulator effectively estimates the value of the held control input by assuming that there are no plant disturbances. In a highly unstable system, the plant disturbances can cause the state to change rapidly resulting in a poor estimate of the held control.

7.5 Mass Ratio Effects

Another parameter of the inverted pendulum system that has a significant effect on regulator performance is the ratio of the mass of the pendulum to the mass of the cart. This parameter was varied to represent a full range of possibilities - from a very light pendulum on a large cart to a heavy pendulum on a small cart. The performance of each of the regulators is listed in Table 7.4. Note that as the mass of the pendulum becomes greater than the mass of the cart, the performance of all regulators rapidly degrades. This characteristic is again due to the increased instability of the system as the mass ratio becomes large. Remember that the pole in the right-half-plane occurs at $\lambda = ((1+r)g/l)^{1/2}$. As r becomes greater than one, the pole location becomes sensitive to it.

With regard to the difference in performance between each of the multirate regulators, it can be seen by examining Table 7.4 that each of the multirate regulators performs very nearly as well as the fast rate regulator over much of the range of r. At large values of r, however, the multirate regulators do begin to show some differences. These differences are illustrated in Figure 7.3 where the percent difference between each multirate regulator and the fast rate regulator are plotted. For small values of r, the performance of the multirate regulators is only a few percent poorer than the fast rate regulator. This is to be expected since a small value of r indicates that the cart is relatively large and sluggish. Little is to be gained by updating the control input to the cart before it has had

Table 7.4 Regulator Performance as a Function of Mass Ratio

(Expected Performance × 10^{-4})

M RATIO	FAST	SLOW	OPTIMAL	OPTIMAL PSO	OPTIMAL CG	OPTIMAL CGPSO	PSO	AVERAGE	SEP MODES
0.005	5.46	12.7	5.73	5.73	5.73	5.73	5.75	5.74	5.75
0.01	5.51	12.8	5.77	5.78	5.78	5.78	5.79	5.78	5.79
0.02	5.61	13.1	5.86	5.86	5.86	5.86	5.87	5.86	5.87
0.05	5.91	13.8	6.13	6.13	6.13	6.13	6.14	6.13	6.14
0.10	6.44	15.1	6.61	6.61	6.61	6.61	6.62	6.61	6.62
0.20	7.60	17.9	7.71	7.71	7.71	7.71	7.72	7.71	7.72
0.50	11.9	28.0	12.2	12.2	12.2	12.2	12.2	12.2	12.2
1.00	22.0	51.1	23.7	23.8	23.9	24.0	24.1	24.3	24.3
2.00	52.8	122	63.4	64.1	65.4	66.6	69.5	68.3	70.7
5.00	231	559	320	323	350	374	486	380	524
10.0	813	2330	1210	1230	1420	1580	3030	1580	4360

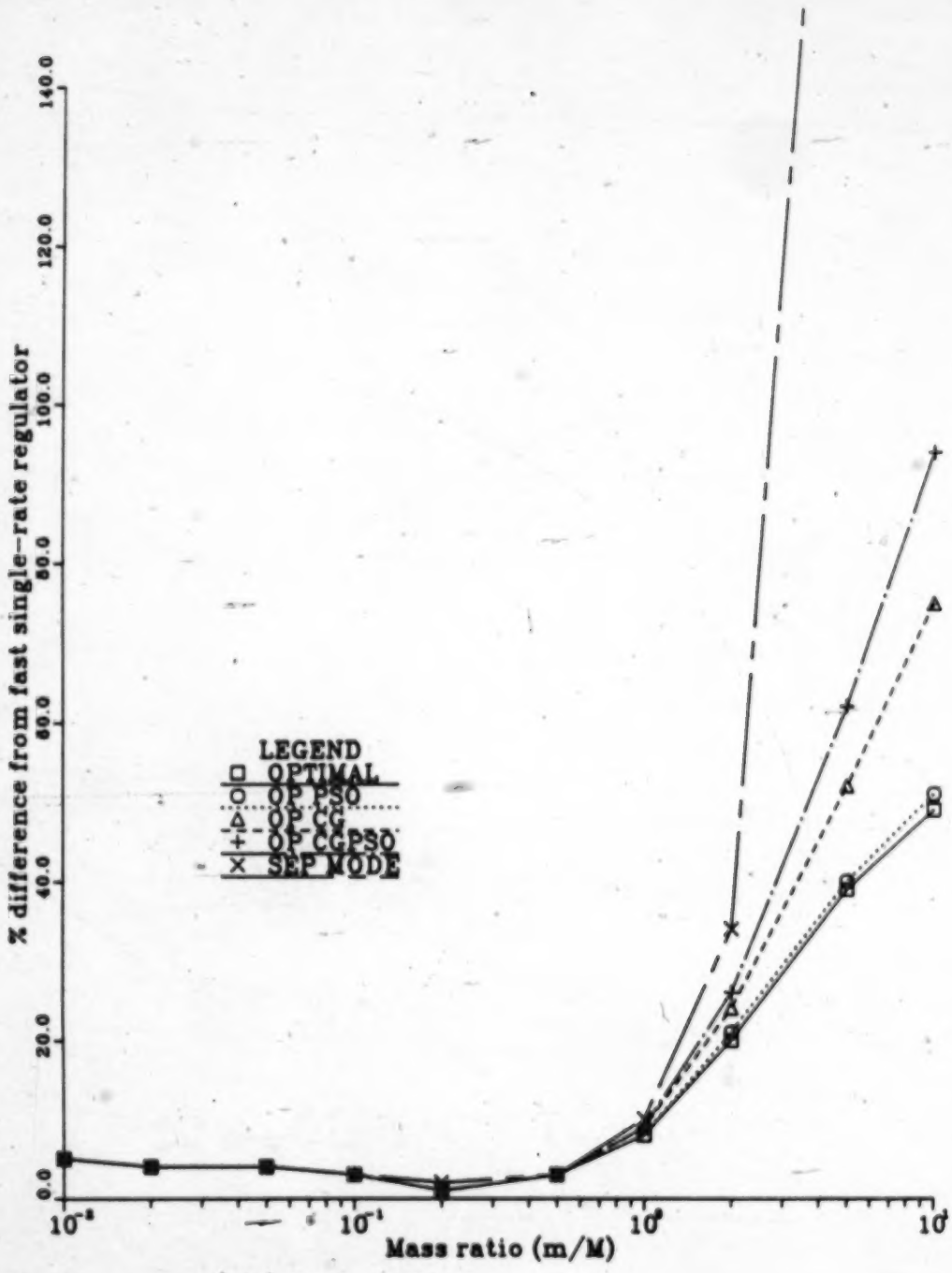

Figure 7.3 - Regulator Performance as a Function of Mass Ratio

time to respond to the previous control input.

For large values of r, the optimal multirate regulator begins to perform somewhat more poorly than the fast rate regulator. In this case, the cart is small and, as a result, rapid updates to its control input are important.

Also interesting for large values of r are the differences between the optimal, optimal constrained, and suboptimal regulators. Here is finally a situation where the optimal multirate regulator performs significantly better than its constrained and suboptimal counterparts. This situation can be understood by considering the characteristics of the system. In the case where $r=10$, the pendulum is ten times more massive than the cart. The cart will move easily when control inputs are applied to it. This motion, of course, will directly and rapidly affect the angle of the pendulum. In other words, the slow control input is strongly coupled into both modes of the system.

The torque input also has a different effect on the system. Due to the fact that the torque input is unable to change the system center of mass and that the pendulum is much heavier than the cart, the torque input will maintain stability by moving the cart under the pendulum (rather than moving the pendulum over the cart). This behaviour corresponds to a large amount of coupling from the fast control input into both modes of the system.

It was initially assumed in this thesis that the multirate regulator would be designed such that the fast control input would primarily drive the fast mode while the

slow control input would primarily drive the slow mode. In the case where the system control inputs are coupled fairly evenly into both system modes, it is not obvious which input should be driven at a faster rate or, in fact, whether multirate control is beneficial.

In spite of the fact that the multirate regulator might not be the best choice when control inputs are strongly coupled to both system modes, it is informative to examine its behaviour in that situation. Consider the response of the torque input when there is a large error in the position of the cart. At the beginning of the control period, the torque regulator knows that a large corrective force will be applied to the cart to restore it to its center position. To keep the pendulum directly above the cart, a large torque input must also be applied. (This corresponds to a negative feedback gain from the cart position.) During the control period, the cart will move closer to its center position and it is expected that during the next control period the force on the cart will be reduced. With the ability to vary its gain, the torquer can anticipate this response and sometime during the control period it can slow the pendulum by applying a torque in the opposite direction. Although the cart is light and can move quickly, the torquer can make a reasonable guess as to its future motions and can begin to compensate for those motions before the cart's control input is updated. Note that, in this case, the feedback gain not only changes in value but also changes sign. This behaviour of the feedback gain from the cart position to the torque control input begins at $m/M \approx 0.5$ and becomes quite

pronounced for $m/M > 5$.

In order for the optimal multirate regulator to compensate for the coupling, two things are required. First, the regulator needs to know the exact value of the control input to the cart. Second, it needs the ability to change its gains during the control period.

7.6 Summary

The design and analysis of a multirate regulator for an unstable system was presented in this chapter. The system consisted of an inverted pendulum mounted on a motorized cart. The state space representation for the system was derived and it was found to have two unstable modes. These modes correspond to the angular motion of the pendulum and the horizontal motion of the cart. A control strategy was proposed in which a horizontal force would be used to control the position of the cart while a high rate torque would be used to control the position of the pendulum.

It was found that over a wide range of the system parameters, the multirate regulator performs very nearly as well as a single-rate regulator that updates both inputs at the fast rate. It was also found that over this wide range of parameters, the constrained and suboptimal multirate regulators perform very nearly as well as the optimal multirate regulator. This would imply that for many situations, the extra effort in solving the optimal multirate regulator and the extra effort needed to implement its complex structure are not necessary.

In cases where the control inputs are strongly coupled into both modes of the system, the time varying gains and held controls become important in the calculation of the optimal controls. In this situation, the slow control has significant influence over the fast mode and in order to choose the best possible fast control, the regulator must know what the slow control input is doing. In addition, the time varying gains of the the fast control allow it to anticipate the actions of the slow control and act accordingly.

Chapter 8

EXAMPLE #2 - THE OSCILLATING CARTS

8.1 Introduction

In the last chapter it was found that, for all practical purposes, the suboptimal multirate regulators performed just as well as the optimal multirate regulator. This was true even over a wide variation of the parameters of the inverted pendulum system. It must be remembered, however, that these results were only from one typical system and, in order to determine their generality, other types of systems must be analyzed.

In this chapter a system having two undamped oscillatory modes will be considered. The physical implementation consists of two carts and three springs but, in order to examine the widest variety of such systems, the parameters in the state equation will be varied over a wide range with little regard for the size of the carts or springs.

The spring-cart system is popular in the literature and some analysis of its characteristics can be found in Housner and Hudson (1959). It was also used extensively by Amit (1980) to examine the characteristics of the optimal multirate regulator and many of his results are duplicated here.

8.2 The Oscillating Carts

Consider the system of two carts of equal mass, m, and three springs of equal spring constant, k, as shown in Figure 8.1. This is the simplest type of system that illustrates the features that appear in a vibration problem when more than one mode of oscillation is present. The control objective is to keep the carts as close to their equilibrium positions as possible. Note that the plant (consisting of carts and springs) is an undamped oscillatory system and any initial displacement of the carts will cause them to oscillate indefinitely.

Plant control is provided by forces applied to each of the carts. In order to illustrate the features of the multirate regulator, these forces contain both slow and fast components. This scheme does not correspond to the characteristics of any particular physical drive motor but is used to examine the characteristics of the general two oscillating mode system while the two carts provide one possible physical implementation of the system equations.

The equations of motion are again obtained by application of Lagrange's equations. The system has two degrees of freedom. The system states and generalized coordinates are chosen to be the displacement of the two carts from their equilibrium positions, x_1 and x_2. The kinetic and potential energies are given by

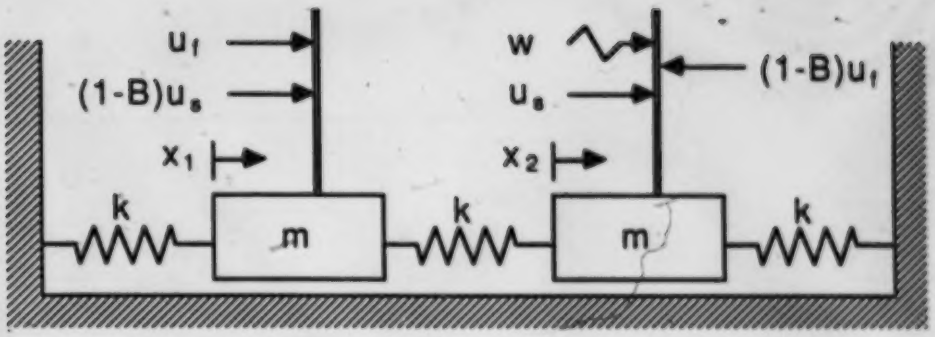

Figure 8.1 - The oscillating carts

$$T = \tfrac{1}{2}m(\dot{x}_1^2 + \dot{x}_2^2)$$
$$V = \tfrac{1}{2}kx_1^2 + \tfrac{1}{2}k(x_1 - x_2)^2 + \tfrac{1}{2}kx_2^2 \qquad (8.2.1)$$
$$= kx_1^2 + kx_2^2 - kx_1x_2.$$

The generalized forces are obtained by determining the amount of work performed during a small displacement of the generalized coordinates. They are given by

$$Q_{x_1} = u_t + (1-\beta)u_s \qquad (8.2.2)$$
$$Q_{x_2} = u_s - (1-\beta)u_t + w.$$

Substituting these expressions into Lagrange's equations gives the two simultaneous differential equations of motion.

$$m\ddot{x}_1 + 2kx_1 - kx_2 = u_t + (1-\beta)u_s \qquad (8.2.3)$$
$$m\ddot{x}_2 + 2kx_2 - kx_1 = u_s - (1-\beta)u_t + w.$$

The linear state space representation can be written

directly.

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \ddot{x}_1 \\ \dot{x}_2 \\ \ddot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ a & 0 & b & 0 \\ 0 & 0 & 0 & 1 \\ b & 0 & a & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ \dot{x}_1 \\ x_2 \\ \dot{x}_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & c \\ 0 & 0 \\ c & 1 \end{bmatrix} \begin{bmatrix} u_f \\ u_s \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} w \qquad (8.2.4)$$

where

$$a = -2k/m \qquad b = k/m \qquad c = -(1-\beta) \qquad (8.2.5)$$

and where u_f, u_s, and w are normalized with respect to m.

From the open-loop characteristic equation,

$$\det(\lambda I - F) = 0, \qquad (8.2.6)$$

the open-loop poles are found to be

$$\lambda = \pm i(-a \pm b)^{1/2}. \qquad (8.2.7)$$

As expected, the system has two undamped oscillatory modes. The natural frequencies of oscillation are

$$\omega = (-a \pm b)^{1/2}. \qquad (8.2.8)$$

By examination of the controllability matrix, it can be seen that the system is controllable through either the fast or slow control inputs as long as $\beta \neq 0$. When $\beta = 0$, the system modes are fully decoupled and the fast control input affects only the fast mode while the slow control input affects only the slow mode.

8.3 Performance of the Multirate Regulators

In his study of the optimal multirate regulator, Amit (1980) provided a fairly detailed analysis of such topics as sample rate selection and effects of system parameters on regulator performance. In order to use those results as a framework for the analysis of this chapter, the same system parameters will be varied while observing the effects on the constrained and suboptimal multirate regulators. The oscillating cart system to be considered has the following characteristics.

1. Slow mode of 1 hertz, i.e., $-a - b = (2\pi)^2$.
2. Fast mode of 8 hertz, i.e., $-a + b = (8*2\pi)^2$.
3. Control period = 0.2 seconds (slow control update rate = 5 hertz).
4. Cycle period = 0.033 seconds (fast control update rate = 30 hertz).
5. Coupling parameter (β) varied between 0 and 1.
6. State to control weighting ratio varied between 5E1 and 5E9.

The performance of the single and multirate regulators for this range of parameters for the oscillating cart system is given in Table 8.1. Note that the separated-modes regulator proposed by Amit is different from that used here. In most situations, the separated-modes regulator used here provides better performance.

Although the oscillating cart system is a totally different type of system than the one considered in the last

Table 8.1 Regulator Performance for the Oscillating Cart System
(Expected Performance × 10^{-7})

W RATIO	β	FAST	SLOW	OPTIMAL	OPTIMAL PSO	OPTIMAL CG	OPTIMAL CGPSO	PSO	AVERAGE	SEP MODES
5E1	1.00	2171	3392	2400	2400	2400	2400	2400	2400	2476
	0.75	2107	3295	2344	2344	2344	2344	2344	2344	2376
	0.50	1944	3050	2200	2200	2200	2200	2200	2200	2210
	0.25	1741	2744	2004	2004	2004	2004	2004	2004	2006
	0.00	1540	2444	1800	1800	1800	1800	1800	1800	1800
5E3	1.00	204.5	487.1	266.4	268.5	268.0	269.2	275.0	268.2	287.8
	0.75	197.7	477.4	303.0	308.2	307.0	309.9	320.7	308.0	327.9
	0.50	180.4	452.8	341.6	348.4	347.4	350.6	360.4	349.7	362.9
	0.25	159.0	422.2	369.0	372.3	371.9	373.2	376.6	373.8	377.0
	0.00	138.1	392.2	368.2	368.2	368.2	368.2	368.2	368.2	368.2
5E5	1.00	14.84	216.5	21.66	21.75	21.71	21.78	43.52	21.73	26.65
	0.75	14.37	215.9	28.61	28.89	28.80	29.03	106.2	28.90	51.59
	0.50	13.20	214.5	44.14	45.05	44.93	45.67	189.0	45.37	130.0
	0.25	11.77	212.9	90.42	93.54	95.60	98.76	232.3	98.36	205.8
	0.00	10.39	211.4	247.8	247.8	247.8	247.8	247.8	247.8	247.8
5E7	1.00	2.612	204.6	8.285	8.394	8.360	8.422	1580	8.421	32.22
	0.75	2.585	204.6	11.02	11.18	11.13	11.21	374.6	11.25	40.40
	0.50	2.517	204.6	17.17	17.47	17.39	17.55	255.7	17.62	---
	0.25	2.437	204.5	37.16	38.18	38.05	38.72	273.7	38.81	229.4
	0.00	2.363	204.5	244.5	244.5	244.5	244.5	244.5	244.5	244.5
5E9	1.00	2.059	204.4	7.986	8.100	8.063	8.126	236,700	8.129	65.27
	0.75	2.058	204.4	10.65	10.80	10.76	10.84	666.1	10.88	69.31
	0.50	2.057	204.4	16.64	16.92	16.84	16.99	1226	17.07	---
	0.25	2.056	204.4	36.07	37.06	36.92	37.57	478.3	37.64	272.2
	0.00	2.055	204.4	244.4	244.4	244.4	244.4	244.4	244.4	244.4

chapter, it does support the conclusions about the characteristics of the multirate regulator. From the performance data of Table 8.1, the following observations can be made.

1. The optimal constrained regulators perform very nearly as well as the optimal multirate regulator even over wide variations of coupling and control weighting. The differences are greatest when the states are weighted much more heavily than the controls and the coupling between modes is large. Even in this case, the difference in performance is only a couple of percent - insignificant for most practical situations.
2. The averaged-gains regulator performs as well as the optimal multirate regulator (again, for all practical purposes) over the full range of parameter variations.
3. The separated-modes regulator performs well until the feedback gains become very large, i.e., when control inputs become inexpensive. When feedback gains (and, as a result, the control inputs) are large, each control input will have a significant effect on both modes of the system. In this case, independent analysis of the fast and slow rate control inputs is not satisfactory.
4. The plant-state-only regulator performs well for moderate values of the state to control weighting ratio. When feedback gains are large, however, the plant-state-only regulator provides somewhat poorer performance than the other optimal and suboptimal regulators.
5. When $\beta=0$ the system modes are fully decoupled and the

fast control drives only the fast mode while the slow control drives only the slow mode. In this situation, single-rate analysis can be independently applied to each of the control inputs and optimal control can be realized without the use of the hold state or time varying gains. This is confirmed by noting that the feedback gains of the optimal and optimal constrained regulators are time invariant and assign a feedback gain of zero to the hold state (see Appendix D). As a result, the optimal, optimal constrained, separated-modes, and averaged-gains regulators are all identical and provide identical performance.

8.4 Summary

The design and analysis of a multirate regulator for the oscillating cart system was presented in this chapter. The state space representation for the system was derived and it was found to have two undamped oscillatory modes. In order to examine the effects of coupling on regulator performance, the control inputs were divided into slow and fast components and the state space representation was parameterized with respect to the coupling between system modes.

It was found that over a wide range of the coupling parameter, the optimal regulator performed very nearly as well as a single-rate regulator. It was also found that, over this wide range of parameters, the constrained and suboptimal multirate regulators performed very nearly as well as the optimal multirate regulator. This supports the

conclusions of the previous chapter. Again, it would seem that the optimal multirate regulator is not necessary in many situations.

The effects of the state to control cost weighting ratio were examined for the first time in this chapter. It was found that over most of the range of this parameter, the multirate regulators were all quite similar in performance. At very large values of this parameter, however, the plant-state-only and separated-modes regulators did not perform as well as the others.

Chapter 9

CONCLUSION

9.1 Summary and Conclusions

The purpose of this thesis was to study the characteristics of the optimal multirate regulator and to search for alternate regulators that are easier to solve and have a simpler structure.

After introducing the problem and the required notation, the design of an optimal multirate regulator was reviewed and an improved algorithm for its solution was developed. The new algorithm is simpler than those published in recent papers as a result of its unique treatment of the additional states needed to hold control values. It relies on the fact that the effects of the fast control inputs can be accumulated over the control period and the system can then be expressed with respect to a single time base. The optimal multirate regulator was found to have two interesting features that are not found in single-rate regulators that control time invariant plants. These are its use of periodically time varying feedback gains and its use of both the plant and held control states in the calculation of new control inputs.

These features, of course, result in a regulator that has a more complicated structure than the standard single-rate regulator. If a simpler structure is to be used for the multirate regulator, it is important to know whether these features are needed to provide optimal control. To

answer this question, a multirate regulator can be considered that is optimal in the sense that it attempts to minimize the same quadratic performance index but is constrained to be time invariant or to use only the plant state in the calculation of new controls. The regulator that is forced to have a time invariant structure is called the optimal constant-gain regulator while the regulator that is not allowed to use the held control state is called the plant-state-only regulator. For interest, a regulator that applies both constraints (the optimal constant-gain plant-state-only regulator) is also considered.

Several interesting results are obtained from the analysis of the optimal contrained regulators. Most important is the fact that, *for all practical purposes, the optimal constrained regulators perform just as well as the optimal multirate regulator over a wide range of the parameters of the system.* The time varying gains and the use of the held control states do become important when there is strong coupling between the fast controls input and both modes of the system or the slow control input and both modes of the system. In this case, it is valuable for the fast control to know what the slow control is doing and to be able to vary its feedback gains to compensate for the effects of the slow control on the fast mode of the system. If optimal performance is important, the designer will be forced to use an optimal multirate regulator for this type of system. Note that in the typical multirate sytem, the fast control input is primarily coupled into the fast system mode and the slow control input is primarily coupled into

the slow system mode. When this is not the case, the regulator problem should be examined in some detail as there is usually nothing to be gained by using a multirate regulator. From this, the following conclusion can be drawn.

> *In multirate systems in which the fast control input is primarily coupled into the fast mode of the plant and the slow control input is primarily coupled into the slow mode of the plant, a constant gain regulator that uses the plant state only can provide near optimal performance.*

Although the optimal constrained regulator provides near optimal performance while utilizing a simpler regulator structure, the numerical methods used in its solution require significant computational resources. While this effort might be justified for large systems in which optimal performance is important, it is often the case that the system designer requires a "quick and dirty" solution that provides good performance but is not necessarily optimal. To meet this need, three suboptimal multirate regulators are proposed and analyzed.

The suboptimal plant-state-only regulator requires solution of the Riccati equation for the equivalent single-rate system but avoids propagating the Riccati equation over the control period by taking a deterministic approach to the solution of the feedback gains. This regulator performs well for stable systems in which plant disturbances are small and for unstable systems in which the

sample rate is fast enough to keep any disturbance from becoming large before being discovered. Although this regulator is adequate in most situations, it requires very nearly as much computation as the optimal multirate regulator and performs more poorly than some of the other multirate regulators.

From the analysis of the optimal multirate regulator, it was found that, in many situations, the feedback gains of the optimal multirate regulator do not vary greatly over the control period. With this in mind, a time invariant regulator that uses the arithmetic mean of the optimal feedback gains should provide good performance. This averaged-gains regulator performs very nearly as well as the optimal multirate regulator over a wide range of the plant parameters. The only situation in which the optimal multirate regulator was observed to perform somewhat better was when the coupling between the plant modes was large. The structure of the averaged-gains regulator is the same as that of the optimal constant-gain regulator and, not surprisingly, their behaviour is found to be quite similar.

To provide a very quick solution to the multirate regulator problem, a separated-modes regulator was considered. This regulator is constructed by designing optimal single-rate regulators for the plant at each of the control update rates. Each control input of the separated-modes regulator then uses the feedback gains of the single-rate regulator having the same sample rate. This is equivalent to assuming that there is no coupling between

system modes, so that single-rate regulators can be independently designed for each of the modes of the system. The separated-modes regulator was found to perform well only as long as the coupling between plant modes was weak.

In light of the results obtained from the analysis and intercomparison of the optimal, optimal constrained, and suboptimal multirate regulators, the following conclusions can be drawn about the structure of the optimal multirate regulator.

The hold state - Slow control inputs are calculated and applied to the plant at the beginning of each control period. Knowledge of these inputs allows the multirate regulator to optimally calculate new controls to assist (or possibly counteract) the slow controls in driving the state to zero. Without knowledge of the slow controls, they appear as unknown external disturbances at all cycles through which they are held. Since the control inputs are a linear combination of the state of the plant, regulators that do not have access to the held control state can estimate its value from the measurements of the plant state. This estimate, however, loses accuracy when the plant state is significantly affected by process noise. From the regulator analysis results, the following conclusion can be drawn.

> *Knowledge of the hold state is important in providing optimal multirate control when the plant disturbances can have a significant effect on the plant state. This occurs when the plant disturbance*

inputs are large or when the sample rate in an unstable plant is slow enough for plant disturbances to become large before being checked.

Time varying gains - From the multirate regulator analysis results, it was found that the optimal multirate regulator uses time varying feedback gains to compensate for strong coupling between system modes. Time varying gains give the fast control input the ability to anticipate the action of the slow control input and to effectively smooth out the effects of the slow control input on the fast mode of the system.

Time varying gains are needed to provide optimal multirate control when plant modes are strongly coupled or when control inputs drive both modes of the plant simultaneously.

9.2 Suggestions for Further Research

Although the design of a multirate regulator was considered in some detail in this thesis, several other topics remain to be considered before a multirate system controller can be constructed. In the previous chapters, it was assumed that an exact measurement of the full plant state was always available. This is rarely true and, in order to control a plant that has outputs sampled at different rates, a multirate state estimator must be considered. Since the multirate regulator and multirate estimator are duals, the results obtained for the multirate regulator can be reinterpreted for the multirate estimator.

This has been done for the Kalman filter by Glasson (1980), but it remains to be done for the estimators that are duals of the other multirate regulators examined here.

Once models have been obtained for the multirate regulator and estimator, they can be combined to form a multirate controller and then realized in one of the many popular forms (see Kailath, 1980). As in the single-rate case, several factors must be considered when choosing a form in which to implement the multirate controller. One important factor is the sensitivity of a particular realization to finite wordlength effects. A good discussion of finite wordlength effects for the single-rate case can be found in Moroney (1983).

The design and analysis of multirate systems using state space techniques is a young and exciting area of research. Until now, however, most of the studies using these techniques have been devoted to designing and analyzing the multirate regulator that minimizes a quadratic performance index (the so called "LQG" problem). Although the solution of this problem has provided insight to the multirate regulator, it seems that such a "high-powered" approach may not always be necessary or even desirable.

One design method that deserves consideration is that of choosing feedback gains to place the system eigenvalues at desired locations. This is commonly known as pole placement design. In conventional time-invariant controller design, there are well known and simple design guidelines that relate pole locations to system performance. It would

be useful if such guidelines could be employed in multirate controller design. Unfortunately, the techniques used in pole placement design of single-rate system controllers are not directly applicable to the multirate case. The main problem exists in defining the closed-loop eigenvalues of the multirate system. Two different definitions can be considered. In the first definition, the slow control input is considered as an external disturbance during all cycles in which it is not updated (the stochastic approach). In the second definition, the process noise effects are ignored and the effects of the slow control input on the deterministic state are included in all cycles (the deterministic approach).

If the slow control input is considered as an external disturbance, it will affect pole locations when it is updated but the poles in all other cycles will only be affected by the fast control input. The weakness of this approach can be seen by considering the plant state trajectory through state space. In linear time-invariant single-rate systems, the plant state asymptotically approaches the origin in state space along a smooth trajectory. If a reference input or constant value disturbance is applied, the plant state will approach a non-zero steady-state value rather than the origin. For the multirate regulator, all control inputs are updated at the beginning of the control period and the plant state trajectory will asymptotically approach the origin. At the beginning of the second cycle, however, the feedback gains change and the slow control input becomes a constant value

disturbance. As a result, the trajectory will change and the state will approach a point that depends on the value of the slow control input. At each of the remaining cycles in the control period the feedback gains can change again and this will cause the state trajectories to also change. Note that the state trajectories are continuous but are no longer smooth. Also, the point that the trajectories approach will change each time a new slow control input is applied and the state trajectories will be continually "chasing" this moving target.

The deterministic approach attempts to avoid the complexity of having the state trajectories chase a moving target. This is done by ignoring the effects of plant disturbances and then including the deterministic effects of the slow control inputs at each cycle. Since the feedback gains are still time-varying, the state trajectories change at the beginning of each cycle but there are no longer external disturbances and all trajectories approach the origin.

In cases where plant disturbances are small, the deterministic approach should provide the expected relationship between pole locations and system response. When plant disturbances are large, however, the slow control input rapidly becomes stale and the stochastic approach might be suitable for cycles in which the slow control is not updated.

It is not clear at this point whether these closed-loop eigenvalue definitions are of any practical value. It could

be that some combination of the two or some totally different definition would provide the expected relationships between closed-loop eigenvalues and system response. Such an investigation could prove to be quite valuable to the advancement of multirate systems theory. †

---

† MULTIRAT presently calculates and prints the deterministic and stochastic eigenvalues of the multirate system at each cycle in the control period.

BIBLIOGRAPHY

Adby, P.R., and M.A.H. Dempster, "Introduction to Optimization Methods", Chapman and Hall, London, 1974.

Amit, N., "Optimal Control of Multirate Digital Control Systems", SUDAAR report #523, Stanford University, July 1980.

Barry, P.E., "Optimal Control of Multirate Systems", Report #RN-361, Research Department, Grumman Aerospace Corp., July 1975.

Brewer, J.W., "Kroenecker Products and Matrix Calculus in System Theory", IEEE Transactions on Circuits and Systems", CAS-25, No. 9, pp 772-781, September 1978.

Broussard, J.R. and N. Halyo, "Optimal Multi-Rate Output Feedback", Proc. 23rd CDC, Vol. II, pp 926-929, Las Vegas NV, December 1984.

Bryson, A.E. and Y.C. Ho, "Applied Optimal Control", Hemisphere Pub. Corp., Washington DC, 1975.

Coffey, T.C. and I.J. Williams, "Stability Analysis of Multiloop, Multirate Sampled Systems", AIAA Journal, Vol. 4, No. 12, pp 2178-2190, December 1966.

Elgerd, O.I., "Control Systems Theory", McGraw-Hill, New York, 1967.

Fortmann, T.E., "An Introduction to Linear Control Systems", Marcel-Dekker, New York, 1977.

Franklin, G.F. and J.D. Powell, "Digital Control of Dynamic Systems", Addison Wesley, Reading Mass., 1980.

Glasson, D.P., "Research in Multirate Estimation and Control", TASC Internal report, Reading Mass., December 1980.

Glasson, D.P., "Development and Applications of Multirate Digital Control", IEEE Control Systems Magazine, pp 2-8, November 1983.

Housner, G.W. and D.E. Hudson, "Applied Mechanics - Dynamics", D. Van Nostrand Co. Inc., Princeton NJ, 1959.

Jury, E.I., "A Note on Multirate Sampled-Data Systems", IEEE Transactions on Automatic Control, AC-12, pp 319-320, June 1967.

Kailath, T., "Linear Systems", Prentice-Hall Inc., Englewood Cliffs NJ, 1980.

Katz, P., "Selection of Sampling Rate for Digital Control of Aircrafts", SUDAAR report #486, Stanford University, September 1974.

Kranc, G.M., "Input-Output Analysis of Multirate Feedback Systems", IRE Transactions on Automatic Control, Vol. PGAC-3, pp 21-28, November 1957.

Kwakernaak, H., and R. Sivan, "Linear Optimal Control Systems", John Wiley & Sons Inc., New York, 1972.

Laub, A.J., "A Schur Method for Solving Algebraic Riccati Equations", IEEE Transactions on Automatic Control, AC-24, No. 6, pp 913-921, December 1979.

Levis, A. and P. Dorato, "Optimal Linear Regulators: The Discrete Time Case", IEEE Transactions on Automatic Control, AC-16, No. 6, pp 613-620, December 1971.

Meyer, R.A. and C.S. Burrus, "A Unified Analysis of Multirate and Periodically Time-Varying Digital Filters", IEEE Transactions on Circuits and Systems, CAS-22, No. 3, pp 162-168, March 1975.

Moroney, P., "Issues in the Impementation of Digital Feedback Compensators", MIT Press, Cambridge Mass, 1983.

Noble, B. and J.W. Daniel, "Applied Linear Algebra", Prentice-Hall, Englewood Cliffs NJ, 1977.

Ragazzini, J.R. and G.F. Franklin, "Sampled-Data Control Systems", McGraw-Hill, 1958.

Sandell, N.R., "Optimal Periodic Sampling Policies for Linear Filtering", TASC Internal Memorandum, Reading Mass., August 1977.

Sklansky, J. and J.R. Ragazzini, "Analysis of Errors in Sampled-Data Systems", AIEE Transactions, Vol. 74, Part II, p 606, May 1955.

Van Loan, C.F., "Computing Integrals Involving the Matrix Exponential", IEEE Transactions on Automatic Control, AC-23, No. 3, pp 395-404, June 1978.

Whitbeck, R.F. and L.G. Hofmann, "Digital Control Law Synthesis in the w' Domain", AIAA J. Guidance and Control, Vol. 1, No. 5, pp 319-326, September 1978.

Whitbeck, R.F. and D.G.J. Didaleusky, "Multirate Digital Control Systems with Simulation Applications", Report No. AFWAL-TR-80-3101, Vol. I, II, III, Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, September 1980.

APPENDIX A

MULTIRAT Description and Listing

MULTIRAT is a software package that provides the tools needed to design and analyze optimal and suboptimal multirate regulators. It consists of a main module and three subroutine libraries. One library contains subroutines that perform the design and analysis of each of the multirate regulators; another contains a collection of matrix manipulation subroutines; and the third contains subroutines that solve the algebraic Riccati equation.

Once started, MULTIRAT calls a replaceable module (named "MODGEN") to generate the continuous-time system matrices from the system parameters (a description of MODGEN and its listings for the "inverted pendulum" and "oscillating cart" systems can be found in Appendix B). The system matrices and dimensions are checked by MULTIRAT and the continuous-time open-loop system eigenvalues are calculated. MULTIRAT then calculates the discrete-time system matrices and the discrete-time open-loop system eigenvalues.

The types of multirate regulators to be analyzed are selected in the main module by adding or deleting calls to the design and analysis subroutines. Once a multirate regulator has been designed and its performance calculated, the results are printed by calling the output subroutine "PERFWR". An example of the data printed by PERFWR can be found in Appendix D.

The types of multirate regulators presently solved and analyzed by MULTIRAT are:

Fast single-rate regulator.

Slow single-rate regulator.

Optimal multirate regulator.

Optimal plant-state-only regulator.

Optimal constant-gain regulator.

Optimal constant-gain plant-state-only regulator.

Suboptimal plant-state-only regulator.

Averaged-gains regulator.

Separated-modes regulator.

For each of these regulators, the following parameters are calculated and printed.

1. Feedback gains for each cycle.
2. Deterministic eigenvalues for each cycle.
3. Stochastic eigenvalues for each cycle.
4. State covariance for each cycle.
5. Expected cost over each cycle.
6. Control period eigenvalues.
7. Total cost over control period.
8. Average cost per cycle.

The MULTIRAT module listings begin on the next page and are followed by the MULSUBS and MATLIB library listings. Note that the IMSL libraries are also used for some matrix manipulations.

```
      PROGRAM MULTIR
C*****************************************************************
C                                                                *
C                                                                *
C                                                                *
C                          MULTIRAT                              *
C                          --------                              *
C                                                                *
C          A Multirate System Design and Analysis Package        *
C                                                                *
C                                                                *
C                                                                *
C----------------------------------------------------------------*
C                                                                *
C                                                                *
C     Written by - Arlen D. SCHMIDT    January 1986              *
C                                                                *
C                                                                *
C----------------------------------------------------------------*
C                                                                *
C                                                                *
C     File - multirat.for                                        *
C                                                                *
C     Limitations - some generalization will be necessary to handle *
C                   more than two control inputs                 *
C                                                                *
C                                                                *
C*****************************************************************
C                                                                *
C                                                                *
C                                                                *
C     MAXIMUM DIMENSIONS WITH PRESENT DECLARATIONS               *
C                                                                *
C             NS=4  NC=2  NG=1  KK=8  LC=9                       *
C                                                                *
C                                                                *
C     ACTUAL MATRIX DIMENSIONS                                   *
C                                                                *
C             F(NS,NS), GU(NS,NC), GW(NS,NG), AA(NS,NS),         *
C             BB(NC,NC), QQ(NG,NG), SA(KK,NC)                    *
C                                                                *
C             PSMAT(KK), FSMAT(KK), PCMAT(KK)                    *
C             WEIGHT(NQ,NQ), FI(NQ,NQ), QD(NS,NS), PHI(NS,NS)    *
C             GAMMA(NS,NC), E(MC,LC), JCYC(KK)                   *
C             PHISLO(NS,NS), GAMSLO(NS,NC), GAISLO(NC,NS)        *
C             WEISLO(NQ,NQ), QDSLO(NS,NS)                        *
C             GAISRS(NC,NS), PCLSRS(NS,NS), RICSRS(NS,NS)        *
C             PHIETI(NS,NS), GAMETI(NS,LC), WEIETI(LCS,LCS)      *
C             GAIETI(LCS,NS), PCLETI(NS,NS), RICETI(NS,NS)       *
```

```
C             PHICYC(FS,PS,KK), GAMCYC(FS,PC,KK)                        *
C             GAICYC(PC,PS,KK), WEICYC(NQ,NQ,KK)                        *
C             COVCYC(PS,PS,KK), GAIAVG(NC,NQ)                           *
C             EVALS(2,NS,KK), SEVALS(2,NS,KK), EIGPER(2,NS)             *
C                                                                       *
C***********************************************************************

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)          *
      PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

      INTEGER NS, NC, NG, KK
      REAL*8 TT
      REAL*8 F(MNS,MNS), GU(MNS,MNC), GW(MNS,MNG)
      REAL*8 AA(MNS,MNS), BB(MNC,MNC), QQ(MNG,MNG), SA(MKK,MNC)

      INTEGER NQ, NS2, NQ2, MC, LC, LCS
      INTEGER PSMAT(MKK), FSMAT(MKK), PCMAT(MKK)
      REAL*8 TO, JTOT, TEMP
      REAL*8 WEIGHT(MNQ,MNQ), FI(MNQ,MNQ), QD(MNS,MNS), PHI(MNS,MNS)
      REAL*8 GAMMA(MNS,MNC), E(MMC,MLC), JCYC(MKK)
      REAL*8 PHISLO(MNS,MNS), GAMSLO(MNS,MNC), GAISLO(MNC,MNS)
      REAL*8 WEISLO(MNQ,MNQ), QDSLO(MNS,MNS)
      REAL*8 GAISRS(MNC,MNS), PCLSRS(MNS,MNS), RICSRS(MNS,MNS)
      REAL*8 PHIETI(MNS,MNS), GAMETI(MNS,MLC), WEIETI(MLCS,MLCS)
      REAL*8 GAIETI(MLCS,MNS), PCLETI(MNS,MNS), RICETI(MNS,MNS)
      REAL*8 PHICYC(MNQ,MNQ,MKK), GAMCYC(MNQ,MNC,MKK)
      REAL*8 GAICYC(MNC,MNQ,MKK), WEICYC(MNQ,MNQ,MKK)
      REAL*8 COVCYC(MNQ,MNQ,MKK), GAIAVG(MNC,MNQ)
      REAL*8 EVALS(2,MNS,MKK), SEVALS(2,MNS,MKK), EIGPER(2,MNS)



C-----------------------------------------------------------------------*
C                                                                       *
C     Generate and echo system description                              *
C                                                                       *
C-----------------------------------------------------------------------*

      WRITE(6,'(15X,''MULTIRATE SYSTEM DESIGN AND ANALYSIS'')')
      WRITE(6,'(15X,''-------- ------ ------ --- --------'',/)')
      WRITE(6,'(''OSYSTEM DESCRIPTION: '',/)')

      CALL MODGEN(F,GU,GW,AA,BB,QQ,SA,TT,KK,NS,NC,NG)
      TO = TT / KK

      WRITE(6,'(''O# of states..... '',I2)') NS
      WRITE(6,'('' # of controls..... '',I2)') NC
      WRITE(6,'('' # of disturbances.. '',I2)') NG
      WRITE(6,'('' # of cycles..... '',I2)') KK
```

```
      WRITE(6,'('' Control period..... '',F10.5)') TT
      WRITE(6,'('' Cycle period..... '',F10.5)') TO

      CALL NORM(SA,KK,NC,TEMP)
      LC = TEMP

      NQ = NS + NC
      NS2 = 2 * NS
      NQ2 = 2 * NQ
      LCS = NS + LC
      MC = KK * NC

      IF ((NS .GT. MNS) .OR. (NC .GT. MNC) .OR. (NG .GT. MNG) .OR.
     *    (KK .GT. MKK) .OR. (LC .GT. MLC)) THEN
        WRITE(6,'(''0*****MULTIRAT - dimensions too large*****'')')
        STOP
      ENDIF

      WRITE(6,'(/,'' The state transition matrix is - '')')
      CALL MWRITE(F,NS,NS)

      WRITE (6,'(/,'' The control distribution matrix is - '')')
      CALL MWRITE(GU,NS,NC)

      WRITE(6,'(/,'' The disturbance distribution matrix is - '')')
      CALL MWRITE(GW,NS,NG)

      WRITE(6,'(/,'' The state weighting matrix is - '')')
      CALL MWRITE(AA,NS,NS)

      WRITE(6,'(/,'' The control weighting matrix is - '')')
      CALL MWRITE(BB,NC,NC)

      WRITE(6,'(/,'' The p.s.d. of the process noise is - '')')
      CALL MWRITE(QQ,NG,NG)

      WRITE(6,'(/,'' The sampling matrix is - '')')
      CALL MWRITE(SA,KK,NC)

      CALL EIGVAL(F,EVALS,NS)
      WRITE(6,'(''0The open loop analog system eigenvalues are:'')')
      DO 60 J = 1,NS
        WRITE(6,'('' '',F9.5,'' + j('',F9.5,'')'')')
     *       EVALS(1,J,1),EVALS(2,J,1)
60    CONTINUE


C-----------------------------------------------------------------------
C                                                                      |
C     Convert from continuous time to discrete time                    |
```

```
C       and calculate necessary parameters                              *
C                                                                       *
C-----------------------------------------------------------------------*

      WRITE(6,'(''0Converting to discrete time.....'')')

      CALL WTCALC(TO,F,GU,AA,BB,WEIGHT,FI,NS,NC,NQ,NQ2)
      CALL QDCALC(TO,F,GW,QQ,QD,PHI,NS,NS2,NG)
      CALL EXTRAC(FI,GAMMA,NQ,NQ,NS,NC,1,NS+1)
      CALL ECALC(E,SA,NC,NC,LC,KK)
      CALL EIGVAL(PHI,EVALS,NS)

      WRITE(6,'(''0The discrete state transition matrix ='')')
      CALL MWRITE(PHI,NS,NS)
      WRITE(6,'(''0The discrete control distribution matrix ='')')
      CALL MWRITE(GAMMA,NS,NC)
      WRITE(6,'(''0The discrete weighting matrix ='')')
      CALL MWRITE(WEIGHT,NQ,NQ)
      WRITE(6,'(''0The discrete process noise covariance ='')')
      CALL MWRITE(QD,NS,NS)
      WRITE(6,'(''0The open loop digital system eigenvalues are:'')')
      DO 70 J = 1,NS
        WRITE(6,'(''    ('',F9.5,'') + j('',F9.5,'')'')')
     *       EVALS(1,J,1),EVALS(2,J,1)
70    CONTINUE

C      WRITE (6,'(/,''0The multirate transformation matrix = '')')
C      CALL MWRITE(E,NC,LC)

      CALL CYCALC(PHI,GAMMA,WEIGHT,SA,PSMAT,FSMAT,PCMAT,
     *            WEICYC,PHICYC,GAMCYC,NS,NC,NQ,KK)


C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the optimal fast single-rate regulator                      *
C                                                                       *
C-----------------------------------------------------------------------*

      CALL GCALC(PHI,GAMMA,WEIGHT,RICSRS,GAISRS,NS,NC,NQ)

      CALL MULT(GAMMA,GAISRS,PCLSRS,NS,NC,NS)
      CALL ADD(PCLSRS,PHI,PCLSRS,NS,NS)
      CALL EIGVAL(PCLSRS,EVALS,NS)
      CALL COCALC(COVCYC,PCLSRS,QD,NS,NS,NS,1)
      CALL JCALC(COVCYC,WEIGHT,GAISRS,JCYC,JTOT,NS,NC,NQ,1)

      WRITE(6,'(''1THE "OPTIMAL FAST SINGLE-RATE" REGULATOR'')')

      WRITE(6,'(''0The fast SRS gain matrix ='')')
```

```
      CALL MWRITE(GAISRS,NC,NS)
      WRITE(6,'(''0The fast SRS closed loop transition matrix ='')')
      CALL MWRITE(PCLSRS,NS,NS)
      WRITE(6,'(''0The fast SRS closed loop eigenvalues are:'')')
      DO 80 J = 1,NS
        WRITE(6,'('' '',F9.5,'' + j('',F9.5,'')'')')
     *       EVALS(1,J,1),EVALS(2,J,1)
80    CONTINUE
      WRITE(6,'(/,''0The fast SRS state covariance ='')')
      CALL MWRITE(COVCYC(1,1,1),NS,NS)
      WRITE(6,'(''0The expected cost per cycle ='',1PE12.5)') JTOT

C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the optimal slow single-rate regulator                      *
C                                                                       *
C-----------------------------------------------------------------------*

      CALL WTCALC(TT,F,GU,AA,BB,WEISLO,FI,NS,NC,NQ,NQ2)
      CALL QDCALC(TT,F,GW,QQ,QDSLO,PHISLO,NS,NS2,NG)
      CALL EXTRAC(FI,GAMSLO,NQ,NQ,NS,NC,1,NS+1)
      CALL EIGVAL(PHISLO,EVALS,NS)

      WRITE(6,'(''1THE "OPTIMAL SLOW SINGLE-RATE" REGULATOR'')')

      WRITE(6,'(''0The slow state transition matrix ='')')
      CALL MWRITE(PHISLO,NS,NS)
      WRITE(6,'(''0The slow control distribution matrix ='')')
      CALL MWRITE(GAMSLO,NS,NC)
      WRITE(6,'(''0The slow process noise covariance ='')')
      CALL MWRITE(QDSLO,NS,NS)
      WRITE(6,'(''0The open loop slow system eigenvalues are:'')')
      DO 90 J = 1,NS
        WRITE(6,'('' '',F9.5,'' + j('',F9.5,'')'')')
     *       EVALS(1,J,1),EVALS(2,J,1)
90    CONTINUE

      CALL GCALC(PHISLO,GAMSLO,WEISLO,RICSRS,GAISLO,NS,NC,NQ)

      CALL MULT(GAMSLO,GAISLO,PCLSRS,NS,NC,NS)
      CALL ADD(PCLSRS,PHISLO,PCLSRS,NS,NS)
      CALL EIGVAL(PCLSRS,EVALS,NS)
      CALL COCALC(COVCYC,PCLSRS,QDSLO,NS,NS,NS,1)
      CALL JCALC(COVCYC,WEISLO,GAISLO,JCYC,JTOT,NS,NC,NQ,1)

      WRITE(6,'(''0The slow SRS gain matrix ='')')
      CALL MWRITE(GAISLO,NC,NS)
      WRITE(6,'(''0The slow SRS closed loop transition matrix ='')')
      CALL MWRITE(PCLSRS,NS,NS)
```

```
      WRITE(6,'(''OThe slow SRS closed loop eigenvalues are:'')')
      DO 95 J = 1,NS
        WRITE(6,'(''    ('',F9.5,'') + j('',F9.5,'')'')')
     *       EVALS(1,J,1),EVALS(2,J,1)
95    CONTINUE
      WRITE(6,'(/,''OThe slow SRS state covariance ='')')
      CALL MWRITE(COVCYC(1,1,1),NS,NS)
      WRITE(6,'('' The expected cost per cycle ='',1PE12.5)') JTOT
      WRITE(6,'(''OExpected cost per cycle / KK ='',1PE12.5)') JTOT/KK


C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the equivalent time invariant (ETI) system                  *
C                                                                       *
C-----------------------------------------------------------------------*

      CALL ETCALC(PHIETI,GAMETI,WEIETI,PHI,GAMMA,WEIGHT,E,
     *            NS,NC,NQ,MC,LC,LCS,KK)

      CALL GCALC(PHIETI,GAMETI,WEIETI,RICETI,GAIETI,NS,LC,LCS)
      CALL MULT(GAMETI,GAIETI,PCLETI,NS,LC,NS)
      CALL ADD(PCLETI,PHIETI,PCLETI,NS,NS)
      CALL EIGVAL(PCLETI,EVALS(1,1,1),NS)


C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the optimal multirate regulator                             *
C                                                                       *
C-----------------------------------------------------------------------*

      CALL OPCALC(RICETI,GAICYC,WEICYC,PHICYC,GAMCYC,
     *            PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      CALL PERFOR(JTOT,JCYC,COVCYC,EVALS,SEVALS,EIGPER,QD,PHICYC,
     *            GAMCYC,GAICYC,WEICYC,PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      CALL AVCALC(GAIAVG,GAICYC,E,PSMAT,PCMAT,NC,MC,LC,NQ,KK)

      WRITE(6,'(''1THE "OPTIMAL MULTIRATE" REGULATOR'')')

      CALL PERFWR(JTOT,JCYC,GAICYC,COVCYC,EVALS,SEVALS,EIGPER,
     *            PSMAT,PCMAT,NS,KK)


C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the optimal plant-state-only regulator                      *
C                                                                       *
```

```
C-----------------------------------------------------------------------*

      CALL CGCALC(QD,PHICYC,GAMCYC,GAICYC,WEICYC,
     *            PSMAT,FSMAT,PCMAT,NS,LC,NQ,KK,NS,.FALSE.)

      CALL PERFOR(JTOT,JCYC,COVCYC,EVALS,SEVALS,EIGPER,QD,PHICYC,
     *            GAMCYC,GAICYC,WEICYC,PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      WRITE(6,'(''1THE "OPTIMAL PLANT-STATE-ONLY" REGULATOR'')')

      CALL PERFWR(JTOT,JCYC,GAICYC,COVCYC,EVALS,SEVALS,EIGPER,
     *            PSMAT,PCMAT,NS,KK)


C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the optimal constant gain regulator                         *
C                                                                       *
C-----------------------------------------------------------------------*

      CALL ASSIGN(GAIAVG,GAICYC,NC,NQ)
      CALL CGCALC(QD,PHICYC,GAMCYC,GAICYC,WEICYC,
     *            PSMAT,FSMAT,PCMAT,NS,NC,NQ,KK,NQ-1,.TRUE.)

      CALL PERFOR(JTOT,JCYC,COVCYC,EVALS,SEVALS,EIGPER,QD,PHICYC,
     *            GAMCYC,GAICYC,WEICYC,PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      WRITE(6,'(''1THE "OPTIMAL CONSTANT GAIN" REGULATOR'')')

      CALL PERFWR(JTOT,JCYC,GAICYC,COVCYC,EVALS,SEVALS,EIGPER,
     *            PSMAT,PCMAT,NS,KK)


C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the optimal constant gain plant-state-only regulator        *
C                                                                       *
C-----------------------------------------------------------------------*

      CALL CGCALC(QD,PHICYC,GAMCYC,GAICYC,WEICYC,
     *            PSMAT,FSMAT,PCMAT,NS,NC,NQ,KK,NS,.TRUE.)

      CALL PERFOR(JTOT,JCYC,COVCYC,EVALS,SEVALS,EIGPER,QD,PHICYC,
     *            GAMCYC,GAICYC,WEICYC,PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      WRITE(6,'(''1THE "OPTIMAL CONSTANT GAIN PSO" REGULATOR'')')

      CALL PERFWR(JTOT,JCYC,GAICYC,COVCYC,EVALS,SEVALS,EIGPER,
     *            PSMAT,PCMAT,NS,KK)
```

```
C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the suboptimal plant-state-only regulator                   *
C                                                                       *
C-----------------------------------------------------------------------*

      CALL PSCALC(PHI,GAMMA,GAIETI,E,SA,GAICYC,PSMAT,NS,NC,MC,LC,KK)

      CALL PERFOR(JTOT,JCYC,COVCYC,EVALS,SEVALS,EIGPER,QD,PHICYC,
     *            GAMCYC,GAICYC,WEICYC,PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      WRITE(6,'(''1THE "SUBOPTIMAL PLANT-STATE-ONLY" REGULATOR'')')

      CALL PERFWR(JTOT,JCYC,GAICYC,COVCYC,EVALS,SEVALS,EIGPER,
     *            PSMAT,PCMAT,NS,KK)


C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the averaged gains regulator                                *
C                                                                       *
C-----------------------------------------------------------------------*

      DO 100 I = 1,KK
        CALL EXTRAC(GAIAVG,GAICYC(1,1,I),NC,NQ,PCMAT(I),PSMAT(I),1,1)
100   CONTINUE

      CALL PERFOR(JTOT,JCYC,COVCYC,EVALS,SEVALS,EIGPER,QD,PHICYC,
     *            GAMCYC,GAICYC,WEICYC,PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      WRITE(6,'(''1THE "AVERAGED GAINS" REGULATOR'')')

      CALL PERFWR(JTOT,JCYC,GAICYC,COVCYC,EVALS,SEVALS,EIGPER,
     *            PSMAT,PCMAT,NS,KK)


C-----------------------------------------------------------------------*
C                                                                       *
C     Solve the multirate regulator using separated modes gains         *
C                                                                       *
C-----------------------------------------------------------------------*

      CALL EXTRAC(GAISLO,GAICYC,NC,NS,1,NS,2,1)
      CALL INSERT(GAISRS,GAICYC,NC,NS,1,NS,2,1)
      DO 110 I = 1,KK
        CALL ZERO(GAICYC(1,1,I),PCMAT(I),PSMAT(I))
        CALL EXTRAC(GAISRS,GAICYC(1,1,I),NC,NS,PCMAT(I),NS,1,1)
110   CONTINUE

```

```
      CALL PERFOR(JTOT,JCYC,COVCYC,EVALS,SEVALS,EIGPER,QD,PHICYC,
     *            GAMCYC,GAICYC,WEICYC,PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      WRITE(6,'(''1THE "SEPARATED MODES" MULTIRATE REGULATOR'')')

      CALL PERFWR(JTOT,JCYC,GAICYC,COVCYC,EVALS,SEVALS,EIGPER,
     *            PSMAT,PCMAT,NS,KK)


      STOP
      END
End of file
```

```
C*****************************************************************
C                                                                *
C                                                                *
C                                                                *
C                          MULSUBS                               *
C                          -------                               *
C                                                                *
C                  Subroutines for use with MULTIRAT             *
C                                                                *
C                                                                *
C                                                                *
C----------------------------------------------------------------*
C                                                                *
C                                                                *
C     written by - Arlen D. Schmidt        January 1986          *
C                                                                *
C                                                                *
C----------------------------------------------------------------*
C                                                                *
C                                                                *
C     file - mulsubs.for                                         *
C                                                                *
C                                                                *
C*****************************************************************



C*****************************************************************
C                                                                *
C                                                                *
C     Calculate equivalent time invariant (ETI)                  *
C       system describing matrices - PHIETI, GMAETI, WEIETI       *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE ETCALC(PHIETI,GMAETI,WEIETI,PHI,GAMMA,WEIGHT,E,
     *                  NS,NC,NQ,MC,LC,LCS,KK)

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
      PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

      INTEGER NS, NC, NQ, MC, LC, LCS, KK
      REAL*8 PHIETI(NS,NS), GMAETI(NS,LC), WEIETI(LCS,LCS)
      REAL*8 PHI(NS,NS), GAMMA(NS,NC), WEIGHT(NQ,NQ), E(MC,LC)

      REAL*8 GMI(MNS,MMC), LMI(MNC,MMC), ZNCNC(MNC,MNC)
      REAL*8 TERM1(MNS,MNC), T1(MNS,MMC)
      REAL*8 TIE(MNQ,MLCS), CINC(MLCS,MLCS)
```

```
51
52      C
53      C          ACTUAL MATRIX DIMENSIONS
54      C
55      C              GMI(NS,MC), LMI(NC,MC), ZNCNC(NC,NC)
56      C              TERM1(NS,NC), T1(NS,MC)
57      C
58            CALL ZERO(WEIETI,LCS,LCS)
59            CALL ZERO(GMI,NS,MC)
60            CALL ZERO(LMI,NC,MC)
61            CALL ZERO(ZNCNC,NC,NC)
62            CALL ZERO(TIE,NQ,LCS)
63
64            CALL IDENT(PHIETI,NS)
65            CALL ASSIGN(GAMMA,TERM1,NS,NC)
66            CALL IDENT(T1,NC)
67            CALL INSERT(LMI,T1,NC,MC,NC,NC,1,1)
68
69            DO 10 I = 1,KK
70
71              CALL INSERT(TIE,PHIETI,NQ,LCS,NS,NS,1,1)
72              CALL MULT(GMI,E,T1,NS,MC,LC)
73              CALL INSERT(TIE,T1,NQ,LCS,NS,LC,1,NS+1)
74              CALL MULT(LMI,E,T1,NC,MC,LC)
75              CALL INSERT(TIE,T1,NQ,LCS,NC,LC,NS+1,NS+1)
76
77              CALL TMULT(TIE,WEIGHT,CINC,NQ,LCS)
78              CALL ADD(CINC,WEIETI,WEIETI,LCS,LCS)
79
80              CALL EXTRAC(GMI,T1,NS,MC,NS,MC-NC,1,1)
81              CALL INSERT(GMI,T1,NS,MC,NS,MC-NC,1,NC+1)
82              CALL INSERT(GMI,TERM1,NS,MC,NS,NC,1,1)
83
84              CALL EXTRAC(LMI,T1,NC,MC,NC,MC-NC,1,1)
85              CALL INSERT(LMI,T1,NC,MC,NC,MC-NC,1,NC+1)
86              CALL INSERT(LMI,ZNCNC,NC,MC,NC,NC,1,1)
87
88              CALL MULT(PHI,TERM1,TERM1,NS,NS,NC)
89              CALL MULT(PHI,PHIETI,PHIETI,NS,NS,NS)
90
91      10    CONTINUE
92
93            CALL MULT(GMI,E,GMAETI,NS,MC,LC)
94
95            RETURN
96            END
97
98
99
100
```

```
101
102     C*************************************************************************
103     C                                                                        *
104     C                                                                        *
105     C     Calculate the E matrix                                             *
106     C       (E is used to transform the full control vector                  *
107     C        to the reduced control vector)                                  *
108     C                                                                        *
109     C                                                                        *
110     C*************************************************************************
111
112           SUBROUTINE ECALC(E,SA,NC,MC,LC,KK)
113
114           PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
115           PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)
116
117           INTEGER NC, MC, LC, KK
118           REAL*8 E(MC,LC), SA(KK,NC)
119
120           INTEGER CNT, CCNT, PNT(MNC)
121           REAL*8 T1(MMC,MMC)
122
123           CALL IDENT(T1,MMC)
124
125           CNT = 1
126           CCNT = 1
127
128     C              * Perform column adding *
129     C              * be careful with the local use of T1 *
130           DO 20 I=1,KK
131             DO 10 J=1,NC
132
133               IF(SA(I,J) .EQ. 0) THEN
134     C                * add column CNT to column PNT(J) *
135                  DO 5 K=1,MC
136                    T1(K,PNT(J)) = T1(K,PNT(J)) + T1(K,CNT)
137     5            CONTINUE
138               ELSE
139                 PNT(J) = CCNT
140                 CCNT = CCNT + 1
141     C              * move column CNT to column PNT(J) *
142                 DO 6 K=1,MC
143                   T1(K,PNT(J)) = T1(K,CNT)
144     6           CONTINUE
145               ENDIF
146               CNT = CNT + 1
147
148     10      CONTINUE
149     20    CONTINUE
150
```

```
151           CALL EXTRAC(T1,E,MMC,MMC,MC,LC,1,1)
152
153           RETURN
154           END
155
156
157
158
159
160     C*************************************************************************
161     C                                                                        *
162     C                                                                        *
163     C     Calculate the discrete process noise covariance                    *
164     C                                                                        *
165     C                                                                        *
166     C*************************************************************************
167
168           SUBROUTINE QDCALC(TO,F,GW,QQ,QD,PHI,NS,NS2,NG)
169
170           PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
171           PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)
172           PARAMETER (MNS2 = 2 * MNS, MNQ2 = 2 * MNQ)
173
174           INTEGER NS, NS2, NG
175           REAL*8 TO
176           REAL*8 F(NS,NS), GW(NS,NG), QQ(NG,NG)
177           REAL*8 QD(NS,NS), PHI(NS,NS)
178
179           REAL*8 WTT1(MNS2,MNS2), WTT2(MNS2,MNS2), WSS1(MNS,MNS)
180
181           CALL MULTT(GW,QQ,PHI,NS,NG)
182           CALL ZERO(WTT1,NS2,NS2)
183           CALL INSERT(WTT1,PHI,NS2,NS2,NS,NS,1,NS+1)
184
185           CALL SCAM(F,WSS1,NS,NS,-1DO)
186           CALL INSERT(WTT1,WSS1,NS2,NS2,NS,NS,1,1)
187
188           CALL TRANS(F,WSS1,NS,NS)
189           CALL INSERT(WTT1,WSS1,NS2,NS2,NS,NS,NS+1,NS+1)
190
191           CALL SCAM(WTT1,WTT1,NS2,NS2,TO)
192           CALL EXP(WTT1,WTT2,NS2)
193
194           CALL EXTRAC(WTT2,WSS1,NS2,NS2,NS,NS,NS+1,NS+1)
195           CALL TRANS(WSS1,PHI,NS,NS)
196           CALL EXTRAC(WTT2,WSS1,NS2,NS2,NS,NS,1,NS+1)
197           CALL MULT(PHI,WSS1,QD,NS,NS,NS)
198
199           RETURN
200           END
```

```
C*************************************************************************
C                                                                        *
C                                                                        *
C     Calculate the discrete system weighting matrices                   *
C                                                                        *
C                                                                        *
C*************************************************************************

      SUBROUTINE WTCALC(TO,F,GU,AA,BB,WEIGHT,FI,NS,NC,NQ,NQ2)

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
      PARAMETER (MNC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)
      PARAMETER (MNS2 = 2 * MNS, MNQ2 = 2 * MNQ)

      INTEGER NS, NC, NQ, NQ2
      REAL*8 TO
      REAL*8 F(NS,NS), GU(NS,NC), AA(NS,NS), BB(NC,NC)
      REAL*8 WEIGHT(NQ,NQ), FI(NQ,NQ)

      REAL*8 WUU1(MNQ2,MNQ2), WUU2(MNQ2,MNQ2)
      REAL*8 WQQ1(MNQ,MNQ), WQQ2(MNQ,MNQ)

      CALL ZERO(WUU1,NQ2,NQ2)
      CALL ZERO(WQQ1,NQ,NQ)
      CALL INSERT(WQQ1,F,NQ,NQ,NS,NS,1,1)
      CALL INSERT(WQQ1,GU,NQ,NQ,NS,NC,1,NS+1)
      CALL INSERT(WUU1,WQQ1,NQ2,NQ2,NQ,NQ,NQ+1,NQ+1)

      CALL SCAM(WQQ1,WQQ1,NQ,NQ,-1DO)
      CALL TRANS(WQQ1,WQQ2,NQ,NQ)
      CALL INSERT(WUU1,WQQ2,NQ2,NQ2,NQ,NQ,1,1)

      CALL INSERT(WUU1,BB,NQ2,NQ2,NC,NC,NS+1,NQ+NS+1)
      CALL INSERT(WUU1,AA,NQ2,NQ2,NS,NS,1,NQ+1)

      CALL SCAM(WUU1,WUU1,NQ2,NQ2,TO)
      CALL EXP(WUU1,WUU2,NQ2)

      CALL EXTRAC(WUU2,FI,NQ2,NQ2,NQ,NQ,NQ+1,NQ+1)
      CALL TRANS(FI,WQQ1,NQ,NQ)
      CALL EXTRAC(WUU2,WQQ2,NQ2,NQ2,NQ,NQ,1,NQ+1)
      CALL MULT(WQQ1,WQQ2,WEIGHT,NQ,NQ,NQ)

      RETURN
      END
```

```
C*****************************************************************************
C                                                                             *
C                                                                             *
C     Calculate the optimal system gain by solving the steady                 *
C        state Riccati equation                                               *
C                                                                             *
C                                                                             *
C*****************************************************************************

      SUBROUTINE GCALC(PHI,GAMMA,WEIGHT,RICSOL,GAIN,N,M,NM)

      PARAMETER (NMAX = 4, MMAX = 10, N2MAX = 2 * NMAX)

      INTEGER N, M, NM
      REAL*8 PHI(N,N), GAMMA(N,M)
      REAL*8 WEIGHT(NM,NM), RICSOL(N,N), GAIN(M,N)

      INTEGER ITYPE(N2MAX), IPVS(NMAX)
      REAL*8 GMAT(NMAX,NMAX), ZMAT(N2MAX,N2MAX), WMAT(N2MAX,N2MAX)
      REAL*8 ER(N2MAX), EI(N2MAX), WORK(NMAX), SCALE(N2MAX)
      REAL*8 TNN1(NMAX,NMAX), TMM1(MMAX,MMAX), TMN1(MMAX,NMAX)

      REAL*8 PHID(NMAX,NMAX), GMAD(NMAX,MMAX), QD(NMAX,NMAX)
      REAL*8 RD(MMAX,MMAX), DEDIAG(MMAX,NMAX), RDINV(MMAX,MMAX)

C             Check dimensions
      IF (N .GT. NMAX) THEN
        WRITE(6,'(''0*****GCALC - N too large*****'')')
        STOP
      ENDIF
      IF (M .GT. MMAX) THEN
        WRITE(6,'(''0*****GCALC - M too large*****'')')
        STOP
      ENDIF

C
C        DIAGONALIZE SYSTEM MATRICES
C
      CALL ASSIGN(GAMMA,GMAD,N,M)
      CALL EXTRAC(WEIGHT,RD,NM,NM,M,M,N+1,N+1)
      CALL INVERT(RD,RDINV,M)

      CALL EXTRAC(WEIGHT,TMN1,NM,NM,M,N,N+1,1)
      CALL MULT(RDINV,TMN1,DEDIAG,M,M,N)
      CALL SCAM(DEDIAG,DEDIAG,M,N,-1D0)
```

```
      CALL EXTRAC(WEIGHT,QD,NM,NM,N,N,1,1)
      CALL EXTRAC(WEIGHT,TMN1,NM,NM,N,M,1,N+1)
      CALL MULT(TMN1,DEDIAG,TNN1,N,M,N)
      CALL ADD(QD,TNN1,QD,N,N,N)

      CALL MULT(GAMMA,DEDIAG,PHID,N,M,N)
      CALL ADD(PHI,PHID,PHID,N,N,N)

C
C         GMAT = GMAD * INV(RD) * GMAD'
C
      CALL MULTT(GMAD,RDINV,GMAT,N,M)

      CALL ASSIGN(PHID,TNN1,N,N)
      CALL ASSIGN(QD,RICSOL,N,N)
      CALL RICDSD(N,N,N,2*N,N,2*N,TNN1,GMAT,RICSOL,ZMAT,WMAT,
     +            ER,EI,WORK,SCALE,ITYPE,IPVS)
C
C   GAIN = -INV(RD+GMAD'*RICSOL*GMAD) * GMAD'*RICSOL*PHID
C
      CALL TMULT(GMAD,RICSOL,TMM1,N,M)
      CALL ADD(RD,TMM1,TMM1,M,M)
      CALL INVERT(TMM1,TMM1,M)
      CALL TRANS(GMAD,TMN1,N,M)
      CALL MULT(TMN1,TMN1,TMN1,M,M,N)
      CALL MULT(TMN1,RICSOL,TMN1,M,N,N)
      CALL MULT(TMN1,PHID,GAIN,M,N,N)
      CALL SCAM(GAIN,GAIN,M,N,-1DO)

C
C         DEDIAGONALIZE THE RESULTING GAIN
C
      CALL ADD(DEDIAG,GAIN,GAIN,M,N)

      RETURN
      END



C*********************************************************************
C                                                                    *
C                                                                    *
C          Calculate the "plant state only" cycle gains              *
C                                                                    *
C                                                                    *
C*********************************************************************

```

```
      SUBROUTINE PSCALC(PHI,GAMMA,GAIETI,E,SA,GAICYC,
     *                  PSMAT,NS,NC,MC,LC,KK)

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
      PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

      INTEGER NS, NC, MC, LC, KK, PSMAT(KK)
      REAL*8 PHI(NS,NS), GAMMA(NS,NC), GAIETI(LC,NS), E(MC,LC)
      REAL*8 SA(KK,NC), GAICYC(MNC,MNQ,KK)

      INTEGER PC
      REAL*8 CMAT(MMC,MNS), PPROD(MNS,MNS), CGAIN(MNC,MNS)
      REAL*8 TNN1(MNS,MNS), TNN2(MNS,MNS)

      CALL MULT(E,GAIETI,CMAT,MC,LC,NS)
      CALL IDENT(PPROD,NS)

      DO 10 I = 1,KK

        CALL EXTRAC(CMAT,CGAIN,MC,NS,NC,NS,(I-1)*NC+1,1)
        CALL MULT(CGAIN,PPROD,CGAIN,NC,NS,NS)
        CALL MULT(GAMMA,CGAIN,TNN1,NS,NC,NS)
        CALL ADD(PHI,TNN1,TNN1,NS,NS)
        CALL INVERT(TNN1,TNN2,NS)
        CALL MULT(PPROD,TNN2,PPROD,NS,NS,NS)

C             eliminate all controls that are held

        PC = NC
        DO 2 J = NC,1,-1
          IF(SA(I,J) .EQ. 0) THEN
            CALL ROWELI(CGAIN,PC,NS,J)
            PC = PC - 1
          ENDIF
2       CONTINUE

        CALL ZERO(GAICYC(1,1,I),PC,PSMAT(I))
        CALL INSERT(GAICYC(1,1,I),CGAIN,PC,PSMAT(I),PC,NS,1,1)

10    CONTINUE

      RETURN
      END
```

```
C*****************************************************************
C                                                                *
C                                                                *
C            Calculate state covariances                         *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE COCALC(COVCYC,PCLCYC,QD,PSMAT,FSMAT,NS,KK)

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
      PARAMETER (MNC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

      PARAMETER (MNSQ = MNS * MNS, MNQQ = MNQ * MNQ)
      PARAMETER (MNSQQ = MNSQ * MNSQ)

      INTEGER NS, KK
      INTEGER PSMAT(KK), FSMAT(KK)
      REAL*8 QD(NS,NS), PCLCYC(MNQ,MNQ,KK), COVCYC(MNQ,MNQ,KK)

      INTEGER IPVT(MNSQ), NSSQ
      REAL*8 COND, UCPHI(MNQQ), UCW(MNQ,MNQ), SCOV(MNQ,MNQ)
      REAL*8 T1(MNSQQ), WORK(MNSQ)

      CALL IDENT(UCPHI,NS)
      CALL ZERO(UCW,NS,NS)

      DO 100 I = 1,KK
        CALL MULT(PCLCYC(1,1,I),UCPHI,UCPHI,FSMAT(I),PSMAT(I),NS)
        CALL MULTT(PCLCYC(1,1,I),UCW,UCW,FSMAT(I),PSMAT(I))
        CALL ADDTOP(UCW,QD,FSMAT(I),FSMAT(I),NS,NS)
100   CONTINUE

      NSSQ = NS * NS
      CALL ASSIGN(UCW,SCOV,NS,NS)
      DO 200 I = 1,NS
        DO 200 J = 1,NS
        DO 200 K = 1,NS
        DO 200 L = 1,NS
          T1(NSSQ*((J-1)*NS+(L-1))+NS*(I-1)+K)
     *      = -UCPHI((L-1)*NS+K) * UCPHI((J-1)*NS+I)
200   CONTINUE

      CALL ADDIDT(T1,NSSQ)
      CALL MLINEQ(NSSQ,NSSQ,NSSQ,1,T1,SCOV,COND,IPVT,WORK)

      DO 110 I = 1,KK
        CALL ASSIGN(SCOV,COVCYC(1,1,I),PSMAT(I),PSMAT(I))
        CALL MULTT(PCLCYC(1,1,I),SCOV,SCOV,FSMAT(I),PSMAT(I))
        CALL ADDTOP(SCOV,QD,FSMAT(I),FSMAT(I),NS,NS)
```

```
451     110   CONTINUE
452
453           RETURN
454           END
455
456
457
458
459
460     C*********************************************************************
461     C                                                                    *
462     C                                                                    *
463     C              Calculate expected value of cost                      *
464     C                                                                    *
465     C                                                                    *
466     C*********************************************************************
467
468           SUBROUTINE JCALC(COVCYC,WEICYC,GAICYC,JCYC,JTOT,PSMAT,PCMAT,NQ,KK)
469
470           PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
471           PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)
472
473           INTEGER NQ, KK
474           INTEGER PSMAT(KK), PCMAT(KK)
475           REAL*8 JTOT, JCYC(KK)
476           REAL*8 COVCYC(MNQ,MNQ,KK), WEICYC(MNQ,MNQ,KK), GAICYC(MNC,MNQ,KK)
477
478           REAL*8 T1(MNQ,MNQ),T2(MNQ,MNQ)
479
480           JTOT = 0
481           DO 100 I = 1,KK
482             CALL IDENT(T1,PSMAT(I))
483             CALL INSERT(T2,T1,NQ,PSMAT(I),PSMAT(I),PSMAT(I),1,1)
484             CALL INSERT(T2,GAICYC(1,1,I),NQ,PSMAT(I),PCMAT(I),PSMAT(I),
485          *              PSMAT(I)+1,1)
486             CALL TMULT(T2,WEICYC(1,1,I),T1,NQ,PSMAT(I))
487             CALL MMULT(T1,COVCYC(1,1,I),T2,PSMAT(I),PSMAT(I),PSMAT(I))
488             CALL TRACE(T2,PSMAT(I),JCYC(I))
489             JTOT = JTOT + JCYC(I)
490
491     100   CONTINUE
492
493           RETURN
494           END
495
496
497
498
499
500     C*********************************************************************
```

```
C                                                                       .
C                                                                       .
C            Calculate cycle matrices                                   .
C                                                                       .
C                                                                       .
C***********************************************************************

      SUBROUTINE CYCALC(PHI,GAMMA,WEIGHT,SA,PSMAT,FSMAT,PCMAT,
     *                  WEICYC,PHICYC,GAMCYC,NS,NC,NQ,KK)

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
      PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

      INTEGER NS, NC, NQ, KK
      REAL*8 PHI(NS,NS), GAMMA(NS,NC), WEIGHT(NQ,NQ), SA(KK,NC)
      INTEGER PSMAT(KK), FSMAT(KK), PCMAT(KK)
      REAL*8 WEICYC(MNQ,MNQ,KK), PHICYC(MNQ,MNQ,KK), GAMCYC(MNQ,MNC,KK)

      INTEGER PS, PC, FS, SAIND
      REAL*8 PHIAUG(MNQ,MNQ), GAMAUG(MNQ,MNC), QAUG(MNQ,MNQ)
      REAL*8 MAUG(MNC,MNQ), RAUG(MNC,MNC)
      REAL*8 PHIRED(MNQ,MNQ), GAMRED(MNQ,MNC), QRED(MNQ,MNQ)
      REAL*8 MRED(MNC,MNQ), RRED(MNC,MNC)
      REAL*8 T1(MNQ,MNC)

C             generate PHIAUG, GAMAUG, QAUG, MAUG, RAUG

      CALL IDENT(T1,NC)
      CALL ZERO(PHIAUG,NQ,NQ)
      CALL INSERT(PHIAUG,T1,NQ,NQ,NC,NC,NS+1,NS+1)
      CALL INSERT(PHIAUG,PHI,NQ,NQ,NS,NS,1,1)
      CALL INSERT(PHIAUG,GAMMA,NQ,NQ,NS,NC,1,NS+1)

      CALL INSERT(GAMAUG,T1,NQ,NC,NC,NC,NS+1,1)
      CALL INSERT(GAMAUG,GAMMA,NQ,NC,NS,NC,1,1)

      CALL ASSIGN(WEIGHT,QAUG,NQ,NQ)
      CALL EXTRAC(WEIGHT,MAUG,NQ,NQ,NC,NQ,NS+1,1)
      CALL EXTRAC(WEIGHT,RAUG,NQ,NQ,NC,NC,NS+1,NS+1)

C             generate PHIRED, GAMRED, QRED, MRED, RRED

      DO 30 I = KK,1,-1

        CALL ASSIGN(PHIAUG,PHIRED,NQ,NQ)
        CALL ASSIGN(GAMAUG,GAMRED,NQ,NC)
        CALL ASSIGN(QAUG,QRED,NQ,NQ)
        CALL ASSIGN(MAUG,MRED,NC,NQ)
        CALL ASSIGN(RAUG,RRED,NC,NC)

```

```
      PS = NQ
      PC = NC
      FS = NQ

      DO 10 J = NC,1,-1

        IF (SA(I,J) .EQ. 1) THEN
          CALL COLELI(PHIRED,FS,PS,NS+J)
          CALL ROWELI(QRED,PS,PS,NS+J)
          CALL COLELI(QRED,PS-1,PS,NS+J)
          CALL COLELI(MRED,PC,PS,NS+J)
          PS = PS - 1
        ELSE
          CALL COLELI(GAMRED,FS,PC,J)
          CALL ROWELI(MRED,PC,PS,J)
          CALL ROWELI(RRED,PC,PC,J)
          CALL COLELI(RRED,PC-1,PC,J)
          PC = PC - 1
        ENDIF

        SAIND = I + 1
        IF (SAIND .EQ. KK+1) SAIND = 1

        IF (SA(SAIND,J) .EQ. 1) THEN
          CALL ROWELI(PHIRED,FS,PS,NS+J)
          CALL ROWELI(GAMRED,FS,PC,NS+J)
          FS = FS - 1
        ENDIF

10    CONTINUE


      PSMAT(I) = PS
      FSMAT(I) = FS
      PCMAT(I) = PC

      CALL ASSIGN(PHIRED,PHICYC(1,1,I),FS,PS)
      CALL ASSIGN(GAMRED,GAMCYC(1,1,I),FS,PC)

      CALL INSERT(WEICYC(1,1,I),QRED,NQ,NQ,PS,PS,1,1)
      CALL INSERT(WEICYC(1,1,I),RRED,NQ,NQ,PC,PC,PS+1,PS+1)
      CALL INSERT(WEICYC(1,1,I),MRED,NQ,NQ,PC,PS,PS+1,1)
      CALL TRANS(MRED,T1,PC,PS)
      CALL INSERT(WEICYC(1,1,I),T1,NQ,NQ,PS,PC,1,PS+1)

30  CONTINUE

      RETURN
      END
```

```
C*********************************************************************
C                                                                    *
C                                                                    *
C          Calculate optimal feedback gains                          *
C                                                                    *
C               - routine will fail if no controls are applied .     *
C                 during any single cycle                            *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE OPCALC(RICETI,GAICYC,WEICYC,PHICYC,GAMCYC,
     *                  PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
      PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

      INTEGER NS, NQ, KK
      INTEGER PSMAT(KK), FSMAT(KK), PCMAT(KK)
      REAL*8 RICETI(NS,NS), GAICYC(MNC,MNQ,KK)
      REAL*8 PHICYC(MNQ,MNQ,KK), GAMCYC(MNQ,MNC,KK), WEICYC(MNQ,MNQ,KK)

      REAL*8 QCYC(MNQ,MNQ), MCYC(MNC,MNQ), RCYC(MNC,MNC)
      REAL*8 LOKMAT(MNQ,MNQ)

      CALL ASSIGN(RICETI,LOKMAT,NS,NS)

      DO 10 I = KK,1,-1

        CALL EXTRAC(WEICYC(1,1,I),QCYC,NQ,NQ,PSMAT(I),PSMAT(I),1,1)
        CALL EXTRAC(WEICYC(1,1,I),MCYC,NQ,NQ,PCMAT(I),PSMAT(I),
     *              PSMAT(I)+1,1)
        CALL EXTRAC(WEICYC(1,1,I),RCYC,NQ,NQ,PCMAT(I),PCMAT(I),
     *              PSMAT(I)+1,PSMAT(I)+1)

        CALL PROPAG(PHICYC(1,1,I),GAMCYC(1,1,I),QCYC,MCYC,RCYC,
     *              GAICYC(1,1,I),LOKMAT,PSMAT(I),PCMAT(I),FSMAT(I))

10    CONTINUE

      RETURN
      END
```

```
C*************************************************************************
C                                                                        *
C                                                                        *
C     Propagate the Riccati equation over one control period             *
C                                                                        *
C                                                                        *
C*************************************************************************

      SUBROUTINE PROPAG(AMAT,BMAT,QMAT,MMAT,RMAT,GAIN,KMAT,PS,PC,FS)

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
      PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

      INTEGER PS, PC, FS
      REAL*8 AMAT(FS,PS), BMAT(FS,PC)
      REAL*8 QMAT(PS,PS), MMAT(PC,PS), RMAT(PC,PC)
      REAL*8 GAIN(PC,PS), KMAT(FS,FS)

      REAL*8 TNN1(MNQ,MNQ), TMM1(MNC,MNC)
      REAL*8 TNM1(MNQ,MNC), TMN1(MNC,MNQ), TMN2(MNC,MNQ)

C
C     Propagate the Ricatti eqn to obtain the optimal time
C     varying feedback gains.
C
C       GAIN(i) = -inv(R+B'K(i+1)B) * (B'K(i+1)A + M)
C
C       K(i) = A'K(i+1)A + Q + (B'K(i+1)A + M)' * GAIN(i)
C

C
C            Calculate gain
C
      CALL TRANS(BMAT,TMN1,FS,PC)
      CALL MULT(TMN1,KMAT,TMN2,PC,FS,FS)
      CALL MULT(TMN2,BMAT,TMM1,PC,FS,PC)
      CALL ADD(RMAT,TMM1,TMM1,PC,PC)
      CALL INVERT(TMM1,TMM1,PC)

      CALL MULT(TMN2,AMAT,TMN1,PC,FS,PS)
      CALL ADD(TMN1,MMAT,TMN1,PC,PS)
      CALL MULT(TMM1,TMN1,GAIN,PC,PC,PS)

      CALL SCAM(GAIN,GAIN,PC,PS,-1D0)

C
C            K(i) = A'K(i+1)A + Q + (TMN1)'*GAIN
C
      CALL TMULT(AMAT,KMAT,KMAT,FS,PS)
      CALL ADD(KMAT,QMAT,KMAT,PS,PS)
```

```
      CALL TRANS(TNN1,TNM1,PC,PS)
      CALL MULT(TNM1,GAIN,TNN1,PS,PC,PS)
      CALL ADD(KMAT,TNN1,KMAT,PS,PS)

      RETURN
      END


C*********************************************************************
C                                                                    *
C                                                                    *
C     Calculate optimal constant gain                                *
C                                                                    *
C             Subroutine will not work with some sampling schemes    *
C             GAICYC(NC,NQ,1) must contain an initial stabilizing    *
C               first guess at the optimal constant gain             *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE CGCALC(QD,PHICYC,GAMCYC,GAICYC,WEICYC,
     *                  PSMAT,FSMAT,PCMAT,NS,NC,NQ,KK,COLDIM,CONSTG)

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
      PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

      PARAMETER (MLCQ = MLC * MNQ)

      INTEGER NS, NC, NQ, KK, COLDIM
      INTEGER PSMAT(KK), FSMAT(KK), PCMAT(KK)
      REAL*8 QD(NS,NS), PHICYC(MNQ,MNQ,KK), GAMCYC(MNQ,MNC,KK)
      REAL*8 GAICYC(MNC,MNQ,KK), WEICYC(MNQ,MNQ,KK)
      LOGICAL CONSTG

      INTEGER ITER, INDEX
      LOGICAL ALTER
      REAL*8 JAY, JPRI, FACTOR, EXFACT
      REAL*8 KMAT(MLCQ), KPMAT(MLCQ)

C     note local access of elements of KMAT and KPMAT
C     ITER determines accuracy of final value
C          - ITER=10 gives 6 significant digits


      ITER = 10
      INDEX = 1
      CALL ZERO(KMAT,NC,NQ)

      IF (CONSTG) THEN
```

```
751             CALL ASSIGN(GAICYC,KMAT,NC,NS)
752           ELSE
753             DO 20 I = 1,KK
754               CALL INSERT(KMAT,GAICYC(1,1,I),NC,NQ,PCMAT(I),NS,
755          *                INDEX,1)
756               INDEX = INDEX + PCMAT(I)
757       20      CONTINUE
758           ENDIF
759
760           EXFACT = 0.5
761           CALL ASSIGN(KMAT,KPMAT,NC,NQ)
762           CALL COST(JAY,KMAT,QD,PHICYC,GAMCYC,GAICYC,WEICYC,
763          *          PSMAT,FSMAT,PCMAT,NS,NC,NQ,KK,CONSTG)
764
765
766           DO 100 I = 1,ITER
767
768             EXFACT = 0.25 * EXFACT
769             FACTOR = 1 + EXFACT
770       C      WRITE(8,'(''0EXFACT = '',1PE12.5)') EXFACT
771
772       30      CONTINUE
773
774             ALTER = .FALSE.
775
776             DO 80 II = 1,NC
777               DO 60 JJ = 1,COLDIM
778
779                 INDEX = (JJ-1)*NC + II
780
781       40        IF (ABS(KMAT(INDEX)) .GE. 1D-6) THEN
782                   KPMAT(INDEX) = FACTOR * KMAT(INDEX)
783                 ELSE
784                   KPMAT(INDEX) = FACTOR * 1D-3
785                 ENDIF
786                 CALL COST(JPRI,KPMAT,QD,PHICYC,GAMCYC,GAICYC,WEICYC,
787          *                PSMAT,FSMAT,PCMAT,NS,NC,NQ,KK,CONSTG)
788
789                 IF (JPRI .GE. JAY) GOTO 50
790                   JAY = JPRI
791                   KMAT(INDEX) = KPMAT(INDEX)
792                   ALTER = .TRUE.
793       C             WRITE(8,'('' Element '',2I3,'' = '',F9.5,
794       C    *              '' J = '',1PE14.7)') II,JJ,KMAT(INDEX),JAY
795                   GOTO 40
796       50        IF (ABS(KMAT(INDEX)) .GE. 1D-6) THEN
797                   KPMAT(INDEX) = (1/FACTOR) * KMAT(INDEX)
798                 ELSE
799                   KPMAT(INDEX) = FACTOR * (-1D-3)
800                 ENDIF
```

```
                CALL COST(JPRI,KPMAT,QD,PHICYC,GAMCYC,GAICYC,WEICYC,
     *                PSMAT,FSMAT,PCMAT,NS,NC,NQ,KK,CONSTG)

                IF (JPRI .GE. JAY) GOTO 55
                  JAY = JPRI
                  KMAT(INDEX) = KPMAT(INDEX)
                  ALTER = .TRUE.
C                  WRITE(8,'('' Element '',2I3,'' = '',F9.5,
C     *                 '' J = '',1PE14.7)') II,JJ,KMAT(INDEX),JAY
                  GOTO 50

55              KPMAT(INDEX) = KMAT(INDEX)

60            CONTINUE
80          CONTINUE

            IF (ALTER) GOTO 30

100     CONTINUE

        RETURN
        END


C***********************************************************************
C                                                                      *
C                                                                      *
C     Calculate cost for one control period for CGCALC subroutine      *
C                                                                      *
C                                                                      *
C***********************************************************************

        SUBROUTINE COST(JTOT,KMAT,QD,PHICYC,GAMCYC,GAICYC,WEICYC,
     *                PSMAT,FSMAT,PCMAT,NS,NC,NQ,KK,CONSTG)

        PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
        PARAMETER (MMC = MNC + MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

        INTEGER NS, NC, NQ, KK
        INTEGER PSMAT(KK), FSMAT(KK), PCMAT(KK)
        REAL*8 JTOT, KMAT(NC,NQ), QD(NS,NS), WEICYC(MNQ,MNQ,KK)
        REAL*8 PHICYC(MNQ,MNQ,KK), GAMCYC(MNQ,MNC,KK), GAICYC(MNC,MNQ,KK)
        LOGICAL CONSTG

        INTEGER INDEX
        REAL*8 JCYC(MKK), PCLCYC(MNQ,MNQ,MKK), COVCYC(MNQ,MNQ,MKK)

        INDEX = 1
        DO 10 I = 1,KK
```

```
            IF (CONSTG) THEN
              CALL EXTRAC(KMAT,GAICYC(1,1,I),NC,NQ,
     *                    PCMAT(I),PSMAT(I),1,1)
            ELSE
              CALL EXTRAC(KMAT,GAICYC(1,1,I),NC,NQ,
     *                    PCMAT(I),PSMAT(I),INDEX,1)
              INDEX = INDEX + PCMAT(I)
            ENDIF

            CALL MMULT(GAMCYC(1,1,I),GAICYC(1,1,I),PCLCYC(1,1,I),
     *                 FSMAT(I),PCMAT(I),PSMAT(I))
            CALL ADD(PHICYC(1,1,I),PCLCYC(1,1,I),PCLCYC(1,1,I),
     *               FSMAT(I),PSMAT(I))
   10     CONTINUE

          CALL COCALC(COVCYC,PCLCYC,QD,PSMAT,FSMAT,NS,KK)
          CALL JCALC(COVCYC,WEICYC,GAICYC,JCYC,JTOT,PSMAT,PCMAT,NQ,KK)

          RETURN
          END


C*********************************************************************
C                                                                    *
C                                                                    *
C     Calculate average of optimal multirate gains                   *
C                                                                    *
C                                                                    *
C*********************************************************************

          SUBROUTINE AVCALC(GAIAVG,GAICYC,E,PSMAT,PCMAT,NC,MC,LC,NQ,KK)

          PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
          PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

          INTEGER NC, MC, LC, NQ, KK
          INTEGER PSMAT(KK), PCMAT(KK)
          REAL*8 GAIAVG(NC,NQ), E(MC,LC), GAICYC(MNC,MNQ,KK)

          INTEGER INDEX
          REAL*8 GTMP1(MMC,MNQ), GTMP2(MNC,MNQ)

          INDEX = 1
          CALL ZERO(GTMP1,LC,NQ)
          DO 10 I = 1,KK
            CALL INSERT(GTMP1,GAICYC(1,1,I),LC,NQ,PCMAT(I),PSMAT(I),INDEX,1)
            INDEX = INDEX + PCMAT(I)
   10     CONTINUE
```

```
      CALL MULT(E,GTMP1,GTMP1,MC,LC,NQ)
      CALL ZERO(GAIAVG,NC,NQ)
      INDEX = 1
      DO 20 I = 1,KK
        CALL EXTRAC(GTMP1,GTMP2,MC,NQ,NC,NQ,INDEX,1)
        CALL ADD(GTMP2,GAIAVG,GAIAVG,NC,NQ)
        INDEX = INDEX + NC
20    CONTINUE

      CALL SCAM(GAIAVG,GAIAVG,NC,NQ,1DO/KK)

      RETURN
      END


C*********************************************************************
C                                                                    *
C                                                                    *
C     Calculate system performance and eigenvalues                   *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE PERFOR(JTOT,JCYC,COVCYC,EVALS,SEVALS,EIGPER,QD,PHICYC,
     *                  GAMCYC,GAICYC,WEICYC,PSMAT,FSMAT,PCMAT,NS,NQ,KK)

      PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
      PARAMETER (MNC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

      INTEGER NS, NQ, KK
      INTEGER PSMAT(KK), FSMAT(KK), PCMAT(KK)
      REAL*8 JTOT, JCYC(KK), QD(NS,NS), EIGPER(2,NS), EVALS(2,MNS,KK)
      REAL*8 PHICYC(MNQ,MNQ,KK), GAMCYC(MNQ,MNC,KK), GAICYC(MNC,MNQ,KK)
      REAL*8 WEICYC(MNQ,MNQ,KK), COVCYC(MNQ,MNQ,KK), SEVALS(2,MNS,KK)

      REAL*8 PCLPER(MNQ,MNQ), PCLOLD(MNQ,MNS), PCLNEW(MNQ,MNS)
      REAL*8 PCLAMO(MNS,MNS), PCLAMN(MNS,MNS), PCLCYC(MNQ,MNQ,MKK)
      REAL*8 T1(MNS,MNS)


C             Generate PCLPER and PCLCYC

      CALL IDENT(PCLPER,NS)
      DO 10 I = 1,KK
        CALL MULT(GAMCYC(1,1,I),GAICYC(1,1,I),PCLCYC(1,1,I),
     *            FSMAT(I),PCMAT(I),PSMAT(I))
        CALL ADD(PHICYC(1,1,I),PCLCYC(1,1,I),PCLCYC(1,1,I),
     *           FSMAT(I),PSMAT(I))
        CALL MULT(PCLCYC(1,1,I),PCLPER,PCLPER,FSMAT(I),PSMAT(I),NS)
```

```
 10    CONTINUE

C              Calculate eigenvalues

       CALL IDENT(PCLOLD,NS)

       DO 20 I = 1,KK
         CALL MULT(PCLCYC(1,1,I),PCLOLD,PCLNEW,FSMAT(I),PSMAT(I),NS)
         CALL EXTRAC(PCLNEW,PCLANN,FSMAT(I),NS,NS,NS,1,1)
         CALL EXTRAC(PCLOLD,PCLANO,PSMAT(I),NS,NS,NS,1,1)
         CALL INVERT(PCLANO,T1,NS)
         CALL MULT(PCLANN,T1,T1,NS,NS,NS)
         CALL EIGVAL(T1,EVALS(1,1,I),NS)
         CALL EXTRAC(PCLCYC(1,1,I),T1,FSMAT(I),PSMAT(I),NS,NS,1,1)
         CALL EIGVAL(T1,SEVALS(1,1,I),NS)
         CALL ASSIGN(PCLNEW,PCLOLD,FSMAT(I),NS)
 20    CONTINUE

       CALL EIGVAL(PCLPER,EIGPER,NS)

C              Calculate performance

       CALL COCALC(COVCYC,PCLCYC,QD,PSMAT,FSMAT,NS,KK)
       CALL JCALC(COVCYC,WEICYC,GAICYC,JCYC,JTOT,PSMAT,PCMAT,NQ,KK)

       RETURN
       END


C*********************************************************************
C                                                                    *
C                                                                    *
C     Print system performance and eigenvalues                       *
C                                                                    *
C                                                                    *
C*********************************************************************

       SUBROUTINE PERFWR(JTOT,JCYC,GAICYC,COVCYC,EVALS,SEVALS,EIGPER,
      *                  PSMAT,PCMAT,NS,KK)

       PARAMETER (MNS = 4, MNC = 2, MNG = 1, MKK = 8, MLC = 9)
       PARAMETER (MMC = MNC * MKK, MNQ = MNS + MNC, MLCS = MLC + MNS)

       INTEGER NS, KK, PSMAT(KK), PCMAT(KK)
       REAL*8 JTOT, JCYC(KK), GAICYC(MNC,MNQ,KK), COVCYC(MNQ,MNQ,KK)
       REAL*8 EVALS(2,MNS,KK), SEVALS(2,MNS,KK), EIGPER(2,NS)

       DO 110 I = 1,KK
```

```
1001              WRITE(6,'(/,''0For cycle '',I2)') I-1
1002              WRITE(6,'(''0The feedback gain ='')')
1003              CALL MWRITE(GAICYC(1,1,I),PCMAT(I),PSMAT(I))
1004              WRITE(6,'('' The deterministic eigenvalues are:'')')
1005
1006              DO 100 J = 1,NS
1007                WRITE(6,'('' ('',F9.5,'') + j('',F9.5,'')'')')
1008             *        EVALS(1,J,I),EVALS(2,J,I)
1009        100   CONTINUE
1010
1011              WRITE(6,'(''0The stochastic eigenvalues are:'')')
1012
1013              DO 105 J = 1,NS
1014                WRITE(6,'('' ('',F9.5,'') + j('',F9.5,'')'')')
1015             *        SEVALS(1,J,I),SEVALS(2,J,I)
1016        105   CONTINUE
1017
1018              WRITE(6,'(''0The state covariance ='')')
1019              CALL MWRITE(COVCYC(1,1,I),PSMAT(I),PSMAT(I))
1020              WRITE(6,'('' The expected cost = '',1PE12.5)') JCYC(I)
1021
1022        110   CONTINUE
1023
1024              WRITE(6,'(///,''0Over one complete control period ....'')')
1025              WRITE(6,'(''0The closed loop eigenvalues are:'')')
1026
1027              DO 120 J = 1,NS
1028                WRITE(6,'('' ('',F9.5,'') + j('',F9.5,'')'')')
1029             *        EIGPER(1,J),EIGPER(2,J)
1030        120   CONTINUE
1031
1032              WRITE(6,'(''0Total cost per control period = '',1PE12.5)') JTOT
1033              WRITE(6,'(''0The average cost per cycle = '',1PE12.5)') JTOT/KK
1034
1035              RETURN
1036              END
End of file
```

```
1
2     C*********************************************************************
3     C                                                                    *
4     C                                                                    *
5     C     MATRIX SUBROUTINE LIBRARY                                      *
6     C                                                                    *
7     C                                                                    *
8     C*********************************************************************
9
10
11    C
12    C     COMMON area presently shared by MULT, EXP, INVERT, EIGVAL
13    C        MULTT, TMULT, ROWELI, COLELI
14    C
15    C      INTEGER I1,I2,I3,NA,NB,NC,MA,MB,MC
16    C      INTEGER IRAY(20),JRAY(20)
17    C      REAL*8 A,B,C,ARAY(20,20),BRAY(20,20),CRAY(20,20)
18    C      COMMON /MLAREA/ A,B,C,ARAY,BRAY,CRAY
19    C      COMMON /MLAREA/ I1,I2,I3,IRAY,JRAY,NA,NB,NC,MA,MB,MC
20    C
21    C
22
23    C*********************************************************************
24    C                                                                    *
25    C                                                                    *
26    C     Compute:  C = A + B                                            *
27    C                                                                    *
28    C                                                                    *
29    C*********************************************************************
30
31          SUBROUTINE ADD(A,B,C,N,M)
32
33          INTEGER N,M
34          REAL*8 A(N,M), B(N,M), C(N,M)
35
36          DO 10 I=1,N
37            DO 10 J=1,M
38              C(I,J) = A(I,J) + B(I,J)
39    10    CONTINUE
40
41          RETURN
42          END
43
44
45
46
47    C*********************************************************************
48    C                                                                    *
49    C                                                                    *
50    C     Compute:  B = A                                                *
```

```
C                                                                      *
C                                                                      *
C**********************************************************************

      SUBROUTINE ASSIGN(A,B,N,M)

      INTEGER N,M
      REAL*8 A(N,M), B(N,M)

      DO 10 I=1,N
        DO 10 J=1,M
          B(I,J) = A(I,J)
10    CONTINUE

      RETURN
      END


C**********************************************************************
C                                                                      *
C                                                                      *
C     Compute: eigenvalue of IN and place them in EVALS                *
C                                                                      *
C             - where IN is (N*N) and EVALS is an (N*1) complex vector *
C             - if eigenvalues cannot be computed a message is printed *
C             - if N > 20 insufficient local storage exists and        *
C               program is stopped                                     *
C                                                                      *
C                                                                      *
C**********************************************************************

      SUBROUTINE EIGVAL(IN,EVALS,N)

      INTEGER N
      REAL*8 IN(N,N),EVALS(2,N)

      INTEGER I1,I2,I3,NA,NB,NC,MA,MB,MC
      INTEGER IRAY(20),JRAY(20)
      REAL*8 A,B,C,ARAY(20,20),BRAY(20,20),CRAY(20,20)
      COMMON /MLAREA/ A,B,C,ARAY,BRAY,CRAY
      COMMON /MLAREA/ I1,I2,I3,IRAY,JRAY,NA,NB,NC,MA,MB,MC

      IF (N .GT. 20) THEN
        WRITE(*,'(''0******EIGVAL - matrix dimension > 20******'',/)')
        STOP
       ENDIF

      CALL ASSIGN(IN,ARAY,N,N)
      CALL EIGRF(ARAY,N,N,0,EVALS,BRAY,N,CRAY,I1)
```

```
      RETURN
      END



C*********************************************************************
C                                                                    *
C                                                                    *
C     Extract:  B from A                                             *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE EXTRAC(A,B,N,M,NN,MM,OUTROW,OUTCOL)

      INTEGER N,NN,M,MM,OUTCOL,OUTROW
      REAL*8 A(N,M), B(NN,MM)

      IF ((OUTROW+NN-1) .GT. N .OR. (OUTCOL+MM-1) .GT. M) THEN
        WRITE(*,'(''0*******EXTRAC - dimension error*******'',/)')
        STOP
      ELSE
        DO 10 I=1,NN
          DO 10 J=1,MM
            B(I,J) = A(OUTROW+I-1,OUTCOL+J-1)
10      CONTINUE
      ENDIF

      RETURN
      END



C*********************************************************************
C                                                                    *
C                                                                    *
C     OUT = Exp(IN)                                                  *
C             could use eigenvalues of IN to verify convergence      *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE EXP(IN,OUT,N)

      INTEGER N
```

```
      REAL*8 IN(N,N), OUT(N,N)

      INTEGER I1,I2,I3,NA,NB,NC,MA,MB,MC
      INTEGER IRAY(20),JRAY(20)
      REAL*8 A,B,C,ARAY(20,20),BRAY(20,20),CRAY(20,20)
      COMMON /MLAREA/ A,B,C,ARAY,BRAY,CRAY
      COMMON /MLAREA/ I1,I2,I3,IRAY,JRAY,NA,NB,NC,MA,MB,MC

      PARAMETER(ITER=50)


      IF (N .GT. 20) THEN
        WRITE(*,'(''0******EXP - matrix dimension > 20******'',/)')
        STOP
        ENDIF

      CALL IDENT(OUT,N)

      DO 10 I=1,ITER
        CALL MMULT(IN,OUT,ARAY,N,N,N)
        CALL SCAM(ARAY,OUT,N,N,1.0D0/(ITER-I+1))
        DO 10 J=1,N
          OUT(J,J) = OUT(J,J) + 1.0
10    CONTINUE

      RETURN
      END




C*********************************************************************
C                                                                    *
C                                                                    *
C     Compute:  Identity Matrix                                      *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE IDENT(MAT,N)

      INTEGER N
      REAL*8 MAT(N,N)

      CALL ZERO(MAT,N,N)
      DO 10 I=1,N
        MAT(I,I) = 1.0
10    CONTINUE
```

```
      RETURN
      END



C*********************************************************************
C                                                                     *
C                                                                     *
C     Insert:  B into A                                               *
C                                                                     *
C                                                                     *
C*********************************************************************

      SUBROUTINE INSERT(A,B,N,M,NN,MM,INROW,INCOL)

      INTEGER N,M,NN,MM,INROW,INCOL
      REAL*8 A(N,M), B(NN,MM)

      IF ((INROW+NN-1) .GT. N .OR. (INCOL+MM-1) .GT. M) THEN
        WRITE(*,'(''0*******INSERT - dimension error*******'',/)')
        STOP
      ELSE
        DO 10 I=1,NN
          DO 10 J=1,MM
            A(I+INROW-1,J+INCOL-1) = B(I,J)
10      CONTINUE
      ENDIF

      RETURN
      END



C*********************************************************************
C                                                                     *
C                                                                     *
C     Compute:  OUT = IN ** (-1)                                      *
C                                                                     *
C               - where IN is (N*N) and OUT is (N*N)                  *
C               - if IN is singular program is stopped                *
C               - if N > 20 insufficient local storage exists and     *
C                 program is stopped                                  *
C                                                                     *
C                                                                     *
C*********************************************************************
```

```
      SUBROUTINE INVERT(IN,OUT,N)

      INTEGER N
      REAL*8 IN(N,N),OUT(N,N)

      INTEGER I1,I2,I3,NA,NB,NC,MA,MB,MC
      INTEGER IRAY(20),JRAY(20)
      REAL*8 A,B,C,ARAY(20,20),BRAY(20,20),CRAY(20,20)
      COMMON /MLAREA/ A,B,C,ARAY,BRAY,CRAY
      COMMON /MLAREA/ I1,I2,I3,IRAY,JRAY,NA,NB,NC,MA,MB,MC

      IF (N .GT. 20) THEN
        WRITE(*,'(''0******INVERT - matrix dimension > 20******'',/)')
        STOP
        ENDIF

      CALL ASSIGN(IN,ARAY,N,N)
      CALL IDENT(OUT,N)
      CALL MLINEQ(N,N,N,N,ARAY,OUT,C,IRAY,BRAY)

      IF (I1 .EQ. 129) THEN
        WRITE(*,'(''0******INVERT - matrix is singular******'',/)')
        ENDIF

      RETURN
      END



C*********************************************************************
C                                                                    *
C                                                                    *
C     Double precision matrix input routine                          *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE MREADP(INMAT,N,M)

      INTEGER N,M
      REAL*8 INMAT(N,M)

      READ(7,*) ((INMAT(I,J),J=1,M),I=1,N)

      RETURN
      END
```

```
C*****************************************************************
C                                                                *
C                                                                *
C     Matrix input routine                                       *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE MREAD(INMAT,N,M)

      INTEGER N,M
      REAL*8 INMAT(N,M)

      READ(5,*) ((INMAT(I,J),J=1,M),I=1,N)

      RETURN
      END




C*****************************************************************
C                                                                *
C                                                                *
C     Compute:  C = A * B                                        *
C                                                                *
C             C can be either A or B                             *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE MULT(INA,INB,OUT,N,M,K)

      INTEGER N,M,K
      REAL*8 INA(N,M),INB(M,K),OUT(N,K)

      INTEGER I1,I2,I3,NA,NB,NC,MA,MB,MC
      INTEGER IRAY(20),JRAY(20)
      REAL*8 A,B,C,ARAY(20,20),BRAY(20,20),CRAY(20,20)
      COMMON /MLAREA/ A,B,C,ARAY,BRAY,CRAY
      COMMON /MLAREA/ I1,I2,I3,IRAY,JRAY,NA,NB,NC,MA,MB,MC

      CALL MMULT(INA,INB,ARAY,N,M,K)
      CALL ASSIGN(ARAY,OUT,N,K)

      RETURN
      END
```

```
C*****************************************************************
C                                                                *
C                                                                *
C     Compute:  C = A * B                                        *
C                                                                *
C               C cannot be A or B                               *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE MMULT(A,B,C,N,M,K)

      INTEGER N,M,K
      REAL*8 A(N,M),B(M,K),C(N,K)

      DO 10 I=1,N
        DO 10 L=1,K
          C(I,L) = 0.0
          DO 10 J=1,M
            C(I,L) = C(I,L) + A(I,J) * B(J,L)
10    CONTINUE

      RETURN
      END



C*****************************************************************
C                                                                *
C                                                                *
C     Matrix output Routine                                      *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE MWRITE(MAT,N,M)

      INTEGER N,M
      REAL*8 MAT(N,M)

      INTEGER ML,MU

      MU = 8
      DO 20 ML = 1,M,8
        IF (MU .GT. M) MU = M
        DO 10 I = 1,N
```

```
401               WRITE(6,'(1X,8F13.6)')(MAT(I,J),J = ML,MU)
402     10      CONTINUE
403             WRITE(6,*)
404             MU = MU + 8
405     20    CONTINUE
406
407           RETURN
408           END
409
410
411
412
413     C*****************************************************************************
414     C                                                                            *
415     C                                                                            *
416     C     Double precision matrix output routine                                 *
417     C                                                                            *
418     C                                                                            *
419     C*****************************************************************************
420
421           SUBROUTINE MWRIDP(OUTMAT,N,M)
422
423           INTEGER N,M
424           REAL*8 OUTMAT(N,M)
425           INTEGER COUNT
426
427           DO 10 I=1,N
428
429           COUNT=1
430
431           IF (M .LE. 3) THEN
432             WRITE(7,*) (OUTMAT(I,J),J=1,M)
433             GO TO 10
434           ENDIF
435
436     1       CONTINUE
437
438             IF (COUNT+3 .GT. M) GOTO 2
439               WRITE(7,*) (OUTMAT(I,J),J=COUNT,COUNT+2)
440               COUNT = COUNT+3
441               GOTO 1
442
443     2       WRITE(7,*) (OUTMAT(I,J),J=COUNT,M)
444             WRITE(7,*)
445
446     10    CONTINUE
447
448           RETURN
449           END
450
```

```
C*****************************************************************
C                                                                *
C                                                                *
C     Compute:  Result = Norm[C]                                 *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE NORM(C,N,M,RESULT)

      INTEGER N,M
      REAL*8 RESULT,C(N,M)

      RESULT = 0.0
      DO 10 I=1,N
        DO 10 J=1,M
          RESULT = RESULT + ABS(C(I,J))
10    CONTINUE

      RETURN
      END



C*****************************************************************
C                                                                *
C                                                                *
C     Compute:  B = [A] * scale                                  *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE SCAM(A,B,N,M,SCALE)

      INTEGER N,M
      REAL*8 SCALE,A(N,M),B(N,M)

      DO 10 I=1,N
        DO 10 J=1,M
        B(I,J) = A(I,J) * SCALE
10    CONTINUE

      RETURN
      END
```

```
C*****************************************************************
C                                                                *
C                                                                *
C     Compute:  B = transpose of A                               *
C       B cannot be A                                            *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE TRANS(A,B,N,M)

      INTEGER N,M
      REAL*8 A(N,M), B(M,N)

      DO 10 I=1,N
        DO 10 J=1,M
          B(J,I) = A(I,J)
10    CONTINUE

      RETURN
      END




C*****************************************************************
C                                                                *
C                                                                *
C     Compute:  A = [0]                                          *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE ZERO(A,N,M)

      INTEGER N,M
      REAL*8 A(N,M)

      DO 10 I=1,N
        DO 10 J=1,M
          A(I,J) = 0.0
10    CONTINUE

      RETURN
      END
```

```
C*********************************************************************
C                                                                    *
C                                                                    *
C     Compute  OUT = IN1' * IN2 * IN1                                *
C        where OUT(M,M) IN1(N,M) IN2(N,N)                            *
C        OUT can be IN1 or IN2                                       *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE TMULT(IN1,IN2,OUT,N,M)

      INTEGER N,M
      REAL*8 IN1(N,M),IN2(N,N),OUT(M,M)

      INTEGER I1,I2,I3,NA,NB,NC,MA,MB,MC
      INTEGER IRAY(20),JRAY(20)
      REAL*8 A,B,C,ARAY(20,20),BRAY(20,20),CRAY(20,20)
      COMMON /MLAREA/ A,B,C,ARAY,BRAY,CRAY
      COMMON /MLAREA/ I1,I2,I3,IRAY,JRAY,NA,NB,NC,MA,MB,MC

      CALL TRANS(IN1,BRAY,N,M)
      CALL MMULT(IN2,IN1,CRAY,N,N,M)
      CALL MMULT(BRAY,CRAY,OUT,M,N,M)

      RETURN
      END



C*********************************************************************
C                                                                    *
C                                                                    *
C     Compute  OUT = IN1 * IN2 * IN1'                                *
C        where OUT(N,N) IN1(N,M) IN2(M,M)                            *
C        OUT can be IN1 or IN2                                       *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE MULTT(IN1,IN2,OUT,N,M)

      INTEGER N,M
```

```
      REAL*8 IN1(N,M),IN2(M,M),OUT(N,N)

      INTEGER I1,I2,I3,NA,NB,NC,MA,MB,MC
      INTEGER IRAY(20),JRAY(20)
      REAL*8 A,B,C,ARAY(20,20),BRAY(20,20),CRAY(20,20)
      COMMON /MLAREA/ A,B,C,ARAY,BRAY,CRAY
      COMMON /MLAREA/ I1,I2,I3,IRAY,JRAY,NA,NB,NC,MA,MB,MC

      CALL TRANS(IN1,BRAY,N,M)
      CALL MMULT(IN1,IN2,CRAY,N,M,M)
      CALL MMULT(CRAY,BRAY,OUT,N,M,N)

      RETURN
      END


C*********************************************************************
C                                                                    *
C                                                                    *
C          Eliminate row 'ROW' from matrix MAT(N*M)                  *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE ROWELI(MAT,N,M,ROW)

      INTEGER N,M,ROW
      REAL*8 MAT(N,M)

      INTEGER I1,I2,I3,NA,NB,NC,MA,MB,MC
      INTEGER IRAY(20),JRAY(20)
      REAL*8 A,B,C,ARAY(20,20),BRAY(20,20),CRAY(20,20)
      COMMON /MLAREA/ A,B,C,ARAY,BRAY,CRAY
      COMMON /MLAREA/ I1,I2,I3,IRAY,JRAY,NA,NB,NC,MA,MB,MC

      IF (ROW .EQ. 1) THEN
        CALL EXTRAC(MAT,ARAY,N,M,N-1,M,2,1)
        CALL INSERT(MAT,ARAY,N-1,M,N-1,M,1,1)
      ELSE IF (ROW .EQ. N) THEN
        CALL EXTRAC(MAT,ARAY,N,M,ROW-1,M,1,1)
        CALL INSERT(MAT,ARAY,N-1,M,ROW-1,M,1,1)
      ELSE IF ((ROW .GT. 1) .AND. (ROW .LT. N)) THEN
        CALL EXTRAC(MAT,ARAY,N,M,ROW-1,M,1,1)
        CALL EXTRAC(MAT,BRAY,N,M,N-ROW,M,ROW+1,1)
        CALL INSERT(MAT,ARAY,N-1,M,ROW-1,M,1,1)
        CALL INSERT(MAT,BRAY,N-1,M,N-ROW,M,ROW,1)
      ELSE
        WRITE(*,'(''0****** ROWELI - dimension error ******'')')
      ENDIF
```

```
651
652           RETURN
653           END
654
655
656
657
658
659       C*************************************************************************
660       C                                                                        *
661       C                                                                        *
662       C              Eliminate column 'COL' from matrix MAT(N*M)               *
663       C                                                                        *
664       C                                                                        *
665       C*************************************************************************
666
667           SUBROUTINE COLELI(MAT,N,M,COL)
668
669           INTEGER N,M,COL
670           REAL*8 MAT(N,M)
671
672           INTEGER I1,I2,I3,NA,NB,NC,MA,MB,MC
673           INTEGER IRAY(20),JRAY(20)
674           REAL*8 A,B,C,ARAY(20,20),BRAY(20,20),CRAY(20,20)
675           COMMON /MLAREA/ A,B,C,ARAY,BRAY,CRAY
676           COMMON /MLAREA/ I1,I2,I3,IRAY,JRAY,NA,NB,NC,MA,MB,MC
677
678           IF ((COL .GT. 0) .AND. (COL .LT. M)) THEN
679             CALL EXTRAC(MAT,ARAY,N,M,N,M-COL,1,COL+1)
680             CALL INSERT(MAT,ARAY,N,M-1,N,M-COL,1,COL)
681           ELSE IF ((COL .LE. 0) .OR. (COL .GT. M)) THEN
682             WRITE(*,'(''0****** COLELI - dimension error ******'')')
683           ENDIF
684
685           RETURN
686           END
687
688
689
690
691
692       C*************************************************************************
693       C                                                                        *
694       C                                                                        *
695       C              Calculate: TR = trace(MAT)        MAT is (N*N)          *
696       C                                                                        *
697       C                                                                        *
698       C*************************************************************************
699
700           SUBROUTINE TRACE(MAT,N,TR)
```

```
      INTEGER N
      REAL*8 TR,MAT(N,N)

      TR = 0

      DO 10 I = 1,N
        TR = TR + MAT(I,I)
10    CONTINUE

      RETURN
      END




C*********************************************************************
C                                                                    *
C                                                                    *
C            Formatted matrix write routine                          *
C                                                                    *
C                                                                    *
C*********************************************************************

      SUBROUTINE FORMWR(MAT,N,M,WIDTH,FDIM,FMT)

      INTEGER N,M,WIDTH,FDIM
      REAL*8 MAT(N,M)
      CHARACTER FMT(FDIM)

      INTEGER ML,MU

      MU = WIDTH
      DO 20 ML = 1,M,WIDTH
        IF (MU .GT. M) MU = M
        DO 10 I = 1,N
          WRITE(6,FMT)(MAT(I,J),J = ML,MU)
10      CONTINUE
        WRITE(6,*)
        MU = MU + WIDTH
20    CONTINUE

      RETURN
      END



C*********************************************************************
```

```
C                                                                      *
C                                                                      *
C     Compute:  A = A + Identity                                       *
C                                                                      *
C                                                                      *
C**********************************************************************

      SUBROUTINE ADDIDT(A,N)

      INTEGER N
      REAL*8 A(N,N)

      DO 10 I=1,N
        A(I,I) = A(I,I) + 1D0
10    CONTINUE

      RETURN
      END



C**********************************************************************
C                                                                      *
C                                                                      *
C     Compute:  IN1 = IN1 + IN2                                        *
C                                                                      *
C               - where IN2 can have smaller dimensions than IN1       *
C                 and is added into the upper left corner              *
C                                                                      *
C                                                                      *
C**********************************************************************

      SUBROUTINE ADDTOP(IN1,IN2,N1,M1,N2,M2)

      INTEGER N1,M1,N2,M2
      REAL*8 IN1(N1,M1),IN2(N2,M2)

      DO 10 I=1,N2
        DO 10 J=1,M2
          IN1(I,J) = IN1(I,J) + IN2(I,J)
10    CONTINUE

      RETURN
      END
End of file
```

APPENDIX B

MODGEN Description and Listings

The continuous-time system matrices are generated from the system parameters in a replaceable module called "MODGEN" (model generator). Since this module differs from one system to the next, it is compiled separately and combined with MULTIRAT at link time. In this way, a different system can be analyzed by altering MODGEN and not worrying about the workings of MULTIRAT.

At present, MODGEN routines exist for both the "inverted pendulum" and "oscillating cart" systems. The parameters for each of these systems (e.g. length of pendulum or mass of cart) are read from an input file. For the "inverted pendulum" system, the input file has the following format.

1. Two comment lines.
2. Number of cycles per control period.
3. Control period (seconds).
4. Length of pendulum.
5. Ratio of pendulum mass to cart mass.
6. Ratio of state weighting to control weighting.
7. Two comment lines.
8. Sampling matrix.

For the "oscillating cart" system, the format is:

1. Two comment lines.
2. Number of cycles per control period.

3. Control period (seconds).
4. Ratio of open-loop mode frequencies.
5. Coupling parameter.
6. Ratio of state weighting to control weighting.
7. Two comment lines.
8. Sampling matrix.

An example input file for the "inverted pendulum" system can be found in Appendix C.

The MODGEN listings for the "inverted pendulum" and "oscillating cart" systems are included on the following pages.

```
C*****************************************************************
C                                                                *
C                                                                *
C                          MODGEN                                *
C                          ------                                *
C                                                                *
C                Generate the state space model for the          *
C                     inverted pendulum system                   *
C                                                                *
C                                                                *
C----------------------------------------------------------------*
C                                                                *
C                                                                *
C     Written by - Arlen D. Schmidt     February 1986            *
C                                                                *
C     File - pendmod.for                                         *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE MODGEN(F,GU,GW,AA,BB,QQ,SA,TT,KK,NS,NC,NG)

      PARAMETER (NNS = 4, NNC = 2, NNG = 1, MKK = 8)

      INTEGER KK, NS, NC, NG
      REAL*8 TT, F(NNS,NNS), GU(NNS,NNC), GW(NNS,NNG)
      REAL*8 AA(NNS,NNS), BB(NNC,NNC), QQ(NNG,NNG), SA(MKK,NNC)

      REAL*8 LENGTH, MASRAT, WEIRAT, STATEW, GRAV

C           Initialize dimensions

      NS = NNS
      NC = NNC
      NG = NNG

C           Read and initialize parameters

      STATEW = 1.0
      GRAV = 9.8

      READ(5,*)
      READ(5,*)
      READ(5,*) KK,TT,LENGTH,MASRAT,WEIRAT

C           Initialize F

      CALL ZERO(F,NS,NS)
      F(1,2) = 1.0
      F(3,4) = 1.0
```

```
      F(2,3) = -MASRAT * GRAV
      F(4,3) = (MASRAT+1) * GRAV / LENGTH

C             Initialize GU

      CALL ZERO(GU,NS,NC)
      GU(2,1) = -MASRAT / LENGTH
      GU(2,2) = MASRAT + 1
      GU(4,1) = (MASRAT + 1) / (LENGTH * LENGTH)
      GU(4,2) = -(MASRAT + 1) / LENGTH

C             Initialize GW

      CALL ZERO(GW,NS,NG)
      GW(4,1) = (MASRAT + 1) / LENGTH

C             Initialize AA

      CALL ZERO(AA,NS,NS)
      AA(1,1) = STATEW
      AA(3,3) = STATEW

C             Initialize BB

      CALL ZERO(BB,NC,NC)
      BB(1,1) = STATEW / WEIRAT
      BB(2,2) = STATEW / WEIRAT

C             Initialize QQ

      QQ(1,1) = 1.0

C             Initialize SA

      READ(5,*)
      READ(5,*)
      CALL MREAD(SA,KK,NC)

C             Print parameters

      WRITE(6,'(''0An inverted pendulum system'')')
      WRITE(6,'(''0Length of pendulum = '',F6.2)') LENGTH
      WRITE(6,'('' Mass ratio  (m/M) = '',1PE10.3)') MASRAT
      WRITE(6,'('' Weighting ratio (A/B) = '',1PE10.3)') WEIRAT


      RETURN
      END
End of file
```

```
C*****************************************************************
C                                                                *
C                                                                *
C                          MODGEN                                *
C                          ------                                *
C                                                                *
C          Generate the state space model for the                *
C               two cart - three spring system                   *
C                                                                *
C                                                                *
C----------------------------------------------------------------*
C                                                                *
C                                                                *
C     Written by - Arlen D. Schmidt     February 1986            *
C                                                                *
C     File - cartmod.for                                         *
C                                                                *
C                                                                *
C*****************************************************************

      SUBROUTINE MODGEN(F,GU,GW,AA,BB,QQ,SA,TT,KK,NS,NC,NG)

      PARAMETER (NNS = 4, NNC = 2, NNG = 1, MKK = 8)

      INTEGER KK, NS, NC, NG
      REAL*8 TT, F(NNS,NNS), GU(NNS,NNC), GW(NNS,NNG)
      REAL*8 AA(NNS,NNS), BB(NNC,NNC), QQ(NNG,NNG), SA(MKK,NNC)

      REAL*8 D, BETA, AB, ASQ, BSQ, WS

C           Initialize dimensions

      NS = NNS
      NC = NNC
      NG = NNG

C           Read parameters

      READ(5,*)
      READ(5,*)
      READ(5,*) KK,TT,D,BETA,AB

C           Initialize F

      WS = 2 * 3.1415926
      ASQ = WS*WS * ((D*D + 1)/2)
      BSQ = WS*WS * ((D*D - 1)/2)
      CALL ZERO(F,NS,NS)
      F(1,2) = 1
      F(3,4) = 1
```

```
      F(2,1) = -ASQ
      F(2,3) = BSQ
      F(4,1) = BSQ
      F(4,3) = -ASQ

C             Initialize GU

      CALL ZERO(GU,NS,NC)
      GU(2,1) = 1
      GU(2,2) = 1-BETA
      GU(4,1) = BETA-1
      GU(4,2) = 1

C             Initialize GW

      CALL ZERO(GW,NS,NG)
      GW(4,1) = 1

C             Initialize AA

      CALL ZERO(AA,NS,NS)
      AA(1,1) = 0.5
      AA(3,3) = 0.5

C             Initialize BB

      CALL ZERO(BB,NC,NC)
      BB(1,1) = 0.5 / AB
      BB(2,2) = 0.5 / AB

C             Initialize QQ

      QQ(1,1) = 1

C             Initialize SA

      READ(5,*)
      READ(5,*)
      CALL MREAD(SA,KK,NC)

C             Print parameters

      WRITE(6,'(''0A two cart - three spring system'')')
      WRITE(6,'(''0The ratio of open loop modes (D) = '',F4.1)') D
      WRITE(6,'('' The coupling parameter (Beta) = '',F6.3)') BETA
      WRITE(6,'('' The weighting ratio (A/B) = '',1PE10.3)') AB


      RETURN
      END
```

APPENDIX C

A Typical Input File

(Typical input file listing begins on next page.)

```
      1     FILE - PEND1.DAT
      2     KK =      TT =       LENGTH =  MASRAT =  WEIRAT =
      3     4         0.2        1.0       0.1       1E3
      4
      5     SA =
      6     1 1
      7     1 0
      8     1 0
      9     1 0
     10
End of file
```

APPENDIX D

A Typical Output File

(Typical output file listing begins on next page.)

```
                MULTIRATE SYSTEM DESIGN AND ANALYSIS
                --------- ------ ------ --- --------


SYSTEM DESCRIPTION:


An inverted pendulum system

Length of pendulum =   1.00
Mass ratio  (m/M) =  1.000E-01
Weighting ratio (A/B) =  1.000E+03

# of states.......  4
# of controls.....  2
# of disturbances..  1
# of cycles.......  4
Control period.....    0.20000
Cycle period.......    0.05000

The state transition matrix is -
     0.0          1.000000     0.0          0.0
     0.0          0.0         -0.980000     0.0
     0.0          0.0          0.0          1.000000
     0.0          0.0         10.780000     0.0


The control distribution matrix is -
     0.0          0.0
    -0.100000     1.100000
     0.0          0.0
     1.100000    -1.100000


The disturbance distribution matrix is -
     0.0
     0.0
     0.0
     1.100000


The state weighting matrix is -
     1.000000     0.0          0.0          0.0
     0.0          0.0          0.0          0.0
     0.0          0.0          1.000000     0.0
     0.0          0.0          0.0          0.0


The control weighting matrix is -
     0.001000     0.0
```

```
          0.0          0.001000

The p.s.d. of the process noise is -
          1.000000

The sampling matrix is -
          1.000000      1.000000
          1.000000      0.0
          1.000000      0.0
          1.000000      0.0

The open loop analog system eigenvalues are:
     (  0.0     ) + j(  0.0     )
     (  0.0     ) + j(  0.0     )
     (  3.28329) + j(  0.0     )
     ( -3.28329) + j(  0.0     )

Converting to discrete time.....

The discrete state transition matrix =
          1.000000      0.050000     -0.001228     -0.000020
          0.0           1.000000     -0.049220     -0.001228
          0.0           0.0           1.013505      0.050225
          0.0           0.0           0.541424      1.013505

The discrete control distribution matrix =
         -0.000125      0.001375
         -0.005022      0.055022
          0.001378     -0.001378
          0.055247     -0.055247

The discrete weighting matrix =
          0.050000      0.001250     -0.000020     -0.000000     -0.000002      0.000023
          0.001250      0.000042     -0.000001     -0.000000     -0.000000      0.000001
         -0.000020     -0.000001      0.050452      0.001261      0.000023     -0.000023
         -0.000000     -0.000000      0.001261      0.000042      0.000001     -0.000001
         -0.000002     -0.000000      0.000023      0.000001      0.000050     -0.000000
          0.000023      0.000001     -0.000023     -0.000001     -0.000000      0.000050

The discrete process noise covariance =
          0.000000      0.000000     -0.000000     -0.000000
          0.000000      0.000000     -0.000001     -0.000025
         -0.000000     -0.000001      0.000051      0.001526
         -0.000000     -0.000025      0.001526      0.061046
```

```
101
102
103       The open loop digital system eigenvalues are:
104          (   1.00000) + j(  0.0     )
105          (   1.00000) + j(  0.0     )
106          (   1.17841) + j(  0.0     )
107          (   0.84860) + j(  0.0     )
108
109
110
111
112       THE "OPTIMAL FAST SINGLE-RATE" REGULATOR
113
114       The fast SRS gain matrix =
115          -13.225744     -5.678149    -32.648042     -7.613162
116          -22.945213     -6.344489     10.652141      1.213885
117
118
119       The fast SRS closed loop transition matrix =
120            0.970101      0.041986      0.017512      0.002603
121           -1.196077      0.679429      0.700861      0.103800
122            0.013394      0.000918      0.953834      0.038060
123            0.536975      0.036814     -1.850797      0.525834
124
125
126       The fast SRS closed loop eigenvalues are:
127          (  0.72900) + j(  0.18480)
128          (  0.72900) + j( -0.18480)
129          (  0.83560) + j(  0.12478)
130          (  0.83560) + j( -0.12478)
131
132
133       The fast SRS state covariance =
134            0.000701      0.000001      0.000931     -0.002785
135            0.000001      0.009937      0.002778      0.003775
136            0.000931      0.002778      0.002203     -0.000126
137           -0.002785      0.003775     -0.000126      0.090392
138
139
140       The expected cost per cycle = 6.43759E-04
141
142
143
144
145       THE "OPTIMAL SLOW SINGLE-RATE" REGULATOR
146
147       The slow state transition matrix =
148            1.000000      0.200000     -0.020314     -0.001335
149
150
```

```
         0.0          1.000000    -0.210393    -0.020314
         0.0          0.0          1.223459     0.214686
         0.0          0.0          2.314320     1.223459


 The slow control distribution matrix =
      -0.002073     0.022073
      -0.021469     0.221469
       0.022802    -0.022802
       0.236155    -0.236155


 The slow process noise covariance =
       0.000000     0.000001    -0.000014    -0.000092
       0.000001     0.000020    -0.000255    -0.001831
      -0.000014    -0.000255     0.003517     0.027885
      -0.000092    -0.001831     0.027885     0.279909


 The open loop slow system eigenvalues are:
    (  1.00000) + j(  0.0    )
    (  1.00000) + j(  0.0    )
    (  1.92834) + j(  0.0    )
    (  0.51858) + j(  0.0    )

 The slow SRS gain matrix =
      -9.386288    -4.494868   -20.942209    -5.722512
     -12.943383    -4.957848     3.453492     0.354652


 The slow SRS closed loop transition matrix =
       0.733759     0.099883     0.099325     0.018355
      -2.665043    -0.001509     1.004048     0.181084
       0.081109     0.010557     0.667189     0.076115
       0.840026     0.109335    -3.446849    -0.211693


 The slow SRS closed loop eigenvalues are:
    (  0.42353) + j(  0.29128)
    (  0.42353) + j( -0.29128)
    (  0.17035) + j(  0.28990)
    (  0.17035) + j( -0.28990)


 The slow SRS state covariance =
       0.001396     0.000033     0.002195    -0.010416
       0.000033     0.031604     0.010049    -0.040812
       0.002195     0.010049     0.010603    -0.001920
      -0.010416    -0.040812    -0.001920     0.408047
```

```
201     The expected cost per cycle = 6.03440E-03
202
203     Expected cost per cycle / KK = 1.50860E-03
204
205
206
207
208
209     THE "OPTIMAL MULTIRATE" REGULATOR
210
211
212     For cycle  0
213
214     The feedback gain =
215        -11.871394     -5.477785    -33.615552     -7.738590
216        -13.080427     -5.005774     2.939510      0.264790
217
218     The deterministic eigenvalues are:
219        (  0.86267) + j(  0.11581)
220        (  0.86267) + j( -0.11581)
221        (  0.77235) + j(  0.14881)
222        (  0.77235) + j( -0.14881)
223
224     The stochastic eigenvalues are:
225        (  0.86267) + j(  0.11581)
226        (  0.86267) + j( -0.11581)
227        (  0.77235) + j(  0.14881)
228        (  0.77235) + j( -0.14881)
229
230     The state covariance =
231          0.000426     -0.000032      0.000665     -0.002345
232         -0.000032      0.007086      0.002547     -0.002028
233          0.000665      0.002547      0.002561     -0.000323
234         -0.002345     -0.002028     -0.000323      0.100982
235
236     The expected cost =  6.52473E-04
237
238
239     For cycle  1
240
241     The feedback gain =
242        -3.897866     -2.602548    -35.708173     -7.952393      0.540749
243
244     The deterministic eigenvalues are:
245        (  0.84567) + j(  0.12371)
246        (  0.84567) + j( -0.12371)
247        (  0.76215) + j(  0.15989)
248        (  0.76215) + j( -0.15989)
249
250     The stochastic eigenvalues are:
```

```
      (  1.00000) + j(  0.0    )
      (  0.77601) + j(  0.14233)
      (  0.77601) + j( -0.14233)
      (  1.00000) + j(  0.0    )

 The state covariance =
      0.000428     0.000046     0.000666    -0.002432    -0.005249
      0.000046     0.005527     0.002299    -0.000446    -0.018904
      0.000666     0.002299     0.002556    -0.000044    -0.010869
     -0.002432    -0.000446    -0.000044     0.100105     0.059224
     -0.005249    -0.018904    -0.010869     0.059224     0.170394

 The expected cost =  6.61825E-04


 For cycle  2

 The feedback gain =
     -6.212029    -3.579307   -35.377734    -7.933901     0.333311

 The deterministic eigenvalues are:
   (  0.73093) + j(  0.18537)
   (  0.73093) + j( -0.18537)
   (  0.83354) + j(  0.12551)
   (  0.83354) + j( -0.12551)

 The stochastic eigenvalues are:
   (  1.00000) + j(  0.0    )
   (  0.77934) + j(  0.14230)
   (  0.77934) + j( -0.14230)
   (  1.00000) + j(  0.0    )

 The state covariance =
      0.000431     0.000020     0.000659    -0.002405    -0.005957
      0.000020     0.004982     0.002272     0.000262    -0.009462
      0.000659     0.002272     0.002567     0.000001    -0.008167
     -0.002405     0.000262     0.000001     0.098924     0.049086
     -0.005957    -0.009462    -0.008167     0.049086     0.170394

 The expected cost =  6.66266E-04


 For cycle  3

 The feedback gain =
     -8.914602    -4.569014   -34.718725    -7.866240     0.147649

 The deterministic eigenvalues are:
   (  0.65946) + j(  0.22713)
   (  0.65946) + j( -0.22713)
```

```
      (  0.82936) + j(  0.12429)
      (  0.82936) + j( -0.12429)

The stochastic eigenvalues are:
      (  1.00000) + j(  0.0    )
      (  0.78432) + j(  0.14117)
      (  0.78432) + j( -0.14117)
      (  1.00000) + j(  0.0    )

The state covariance =
     0.000430    -0.000033     0.000658    -0.002345    -0.006190
    -0.000033     0.005497     0.002387    -0.000226     0.000118
     0.000658     0.002387     0.002573    -0.000137    -0.006009
    -0.002345    -0.000226    -0.000137     0.098959     0.037413
    -0.006190     0.000118    -0.006009     0.037413     0.170394

The expected cost =  6.63278E-04



Over one complete control period ....

The closed loop eigenvalues are:
      (  0.42152) + j(  0.28693)
      (  0.42152) + j( -0.28693)
      (  0.17306) + j(  0.28361)
      (  0.17306) + j( -0.28361)

Total cost per control period =  2.64394E-03

The average cost per cycle =  6.60986E-04



THE "OPTIMAL PLANT-STATE-ONLY" REGULATOR

For cycle  0

The feedback gain =
   -11.957154    -5.487320   -33.582902    -7.744336
   -13.378375    -5.042600     3.056271     0.246602

The deterministic eigenvalues are:
      (  0.86123) + j(  0.11717)
      (  0.86123) + j( -0.11717)
      (  0.77289) + j(  0.15006)
```

```
      (  0.77289) + j( -0.15006)

The stochastic eigenvalues are:
   (  0.86123) + j(  0.11717)
   (  0.86123) + j( -0.11717)
   (  0.77289) + j(  0.15006)
   (  0.77289) + j( -0.15006)

The state covariance =
     0.000422    -0.000032     0.000662    -0.002339
    -0.000032     0.007026     0.002538    -0.002077
     0.000662     0.002538     0.002565    -0.000320
    -0.002339    -0.002077    -0.000320     0.101098

The expected cost =  6.52682E-04


For cycle  1

The feedback gain =
   -16.629592    -6.059534   -31.595347    -7.960020     0.0

The deterministic eigenvalues are:
   (  0.83671) + j(  0.13780)
   (  0.83671) + j( -0.13780)
   (  0.76326) + j(  0.12853)
   (  0.76326) + j( -0.12853)

The stochastic eigenvalues are:
   (  1.00000) + j(  0.0    )
   (  0.78811) + j(  0.10491)
   (  0.78811) + j( -0.10491)
   (  1.00000) + j(  0.0    )

The state covariance =
     0.000424     0.000046     0.000663    -0.002425    -0.005204
     0.000046     0.005487     0.002293    -0.000495    -0.018713
     0.000663     0.002293     0.002560    -0.000044    -0.010788
    -0.002425    -0.000495    -0.000044     0.100197     0.058549
    -0.005204    -0.018713    -0.010788     0.058549     0.167815

The expected cost =  6.62114E-04


For cycle  2

The feedback gain =
   -16.700494    -5.807550   -31.789904    -8.020716     0.0

The deterministic eigenvalues are:
```

```
401          (  0.82477) + j(  0.13762)
402          (  0.82477) + j( -0.13762)
403          (  0.72722) + j(  0.15576)
404          (  0.72722) + j( -0.15576)
405
406       The stochastic eigenvalues are:
407          (  1.00000) + j(  0.0    )
408          (  0.78567) + j(  0.10240)
409          (  0.78567) + j( -0.10240)
410          (  1.00000) + j(  0.0    )
411
412       The state covariance =
413            0.000428     0.000020     0.000656    -0.002398    -0.005906
414            0.000020     0.004950     0.002267     0.000206    -0.009396
415            0.000656     0.002267     0.002571    -0.000001    -0.008121
416           -0.002398     0.000206    -0.000001     0.099007     0.048357
417           -0.005906    -0.009396    -0.008121     0.048357     0.167815
418
419       The expected cost =  6.66325E-04
420
421
422       For cycle  3
423
424       The feedback gain =
425          -13.775597    -5.265422   -33.185755    -7.926749     0.0
426
427       The deterministic eigenvalues are:
428          (  0.64346) + j(  0.20568)
429          (  0.64346) + j( -0.20568)
430          (  0.83066) + j(  0.13254)
431          (  0.83066) + j( -0.13254)
432
433       The stochastic eigenvalues are:
434          (  1.00000) + j(  0.0    )
435          (  0.78576) + j(  0.12290)
436          (  0.78576) + j( -0.12290)
437          (  1.00000) + j(  0.0    )
438
439       The state covariance =
440            0.000427    -0.000032     0.000654    -0.002339    -0.006139
441           -0.000032     0.005459     0.002381    -0.000287     0.000060
442            0.000654     0.002381     0.002577    -0.000138    -0.006001
443           -0.002339    -0.000287    -0.000138     0.099064     0.036641
444           -0.006139     0.000060    -0.006001     0.036641     0.167815
445
446       The expected cost =  6.63289E-04
447
448
449
450
```

```
451     Over one complete control period ....
452
453     The closed loop eigenvalues are:
454        (  0.40421) + j(  0.29529)
455        (  0.40421) + j( -0.29529)
456        (  0.18404) + j(  0.25818)
457        (  0.18404) + j( -0.25818)
458
459     Total cost per control period =  2.64441E-03
460
461     The average cost per cycle =  6.61102E-04
462
463
464
465
466
467     THE "OPTIMAL CONSTANT GAIN" REGULATOR
468
469
470     For cycle  0
471
472     The feedback gain =
473        -13.211248    -5.401632   -32.961903    -7.904985
474        -13.258497    -5.026023     3.005246     0.244764
475
476     The deterministic eigenvalues are:
477        (  0.86242) + j(  0.12124)
478        (  0.86242) + j( -0.12124)
479        (  0.76818) + j(  0.13160)
480        (  0.76818) + j( -0.13160)
481
482     The stochastic eigenvalues are:
483        (  0.86242) + j(  0.12124)
484        (  0.86242) + j( -0.12124)
485        (  0.76818) + j(  0.13160)
486        (  0.76818) + j( -0.13160)
487
488     The state covariance =
489         0.000421    -0.000030     0.000660    -0.002351
490        -0.000030     0.006994     0.002530    -0.002095
491         0.000660     0.002530     0.002565    -0.000299
492        -0.002351    -0.002095    -0.000299     0.101560
493
494     The expected cost =  6.59758E-04
495
496
497     For cycle  1
498
499     The feedback gain =
500        -13.211248    -5.401632   -32.961903    -7.904985     0.085666
```

```
501
502     The deterministic eigenvalues are:
503        (  0.83997) + j(  0.12801)
504        (  0.83997) + j( -0.12801)
505        (  0.76166) + j(  0.14669)
506        (  0.76166) + j( -0.14669)
507
508     The stochastic eigenvalues are:
509        (  1.00000) + j(  0.0    )
510        (  0.78682) + j(  0.12281)
511        (  0.78682) + j( -0.12281)
512        (  1.00000) + j(  0.0    )
513
514     The state covariance =
515          0.000423     0.000045     0.000661    -0.002410    -0.005181
516          0.000045     0.005447     0.002288    -0.000366    -0.018576
517          0.000661     0.002288     0.002564    -0.000008    -0.010760
518         -0.002410    -0.000366    -0.000008     0.099306     0.057547
519         -0.005181    -0.018576    -0.010760     0.057547     0.166890
520
521     The expected cost =  6.62174E-04
522
523
524     For cycle  2
525
526     The feedback gain =
527        -13.211248    -5.401632   -32.961903    -7.904985     0.085666
528
529     The deterministic eigenvalues are:
530        (  0.73080) + j(  0.17683)
531        (  0.73080) + j( -0.17683)
532        (  0.82600) + j(  0.12656)
533        (  0.82600) + j( -0.12656)
534
535     The stochastic eigenvalues are:
536        (  1.00000) + j(  0.0    )
537        (  0.78682) + j(  0.12281)
538        (  0.78682) + j( -0.12281)
539        (  1.00000) + j(  0.0    )
540
541     The state covariance =
542          0.000427     0.000019     0.000655    -0.002387    -0.005879
543          0.000019     0.004933     0.002269     0.000182    -0.009351
544          0.000655     0.002269     0.002574    -0.000033    -0.008131
545         -0.002387     0.000182    -0.000033     0.098898     0.047856
546         -0.005879    -0.009351    -0.008131     0.047856     0.166890
547
548     The expected cost =  6.61907E-04
549
550
```

```
551      For cycle  3
552
553      The feedback gain =
554         -13.211248    -5.401632   -32.961903    -7.904985     0.085666
555
556      The deterministic eigenvalues are:
557         (  0.66350) + j(  0.22059)
558         (  0.66350) + j( -0.22059)
559         (  0.82010) + j(  0.12365)
560         (  0.82010) + j( -0.12365)
561
562      The stochastic eigenvalues are:
563         (  1.00000) + j(  0.0    )
564         (  0.78682) + j(  0.12281)
565         (  0.78682) + j( -0.12281)
566         (  1.00000) + j(  0.0    )
567
568      The state covariance =
569           0.000426    -0.000031     0.000653    -0.002339    -0.006112
570          -0.000031     0.005448     0.002379    -0.000391     0.000012
571           0.000653     0.002379     0.002577    -0.000163    -0.006023
572          -0.002339    -0.000391    -0.000163     0.099660     0.036658
573          -0.006112     0.000012    -0.006023     0.036658     0.166890
574
575      The expected cost =  6.60749E-04
576
577
578
579
580      Over one complete control period ....
581
582      The closed loop eigenvalues are:
583         (  0.40278) + j(  0.28591)
584         (  0.40278) + j( -0.28591)
585         (  0.18935) + j(  0.27024)
586         (  0.18935) + j( -0.27024)
587
588      Total cost per control period =  2.64459E-03
589
590      The average cost per cycle =  6.61147E-04
591
592
593
594
595
596      THE "OPTIMAL CONSTANT GAIN PSO" REGULATOR
597
598
599      For cycle  0
600
```

```
601      The feedback gain =
602         -15.139446     -5.764192    -32.346949     -7.922675
603         -13.428587     -5.042188      3.063843      0.237380
604
605      The deterministic eigenvalues are:
606            (  0.85953) + j(  0.12604)
607            (  0.85953) + j( -0.12604)
608            (  0.77164) + j(  0.12408)
609            (  0.77164) + j( -0.12408)
610
611      The stochastic eigenvalues are:
612            (  0.85953) + j(  0.12604)
613            (  0.85953) + j( -0.12604)
614            (  0.77164) + j(  0.12408)
615            (  0.77164) + j( -0.12408)
616
617      The state covariance =
618           0.000420     -0.000032      0.000659     -0.002337
619          -0.000032      0.006982      0.002528     -0.002135
620           0.000659      0.002528      0.002567     -0.000297
621          -0.002337     -0.002135     -0.000297      0.101530
622
623      The expected cost =  6.60024E-04
624
625
626      For cycle  1
627
628      The feedback gain =
629         -15.139446     -5.764192    -32.346949     -7.922675      0.0
630
631      The deterministic eigenvalues are:
632            (  0.84182) + j(  0.13250)
633            (  0.84182) + j( -0.13250)
634            (  0.75837) + j(  0.13900)
635            (  0.75837) + j( -0.13900)
636
637      The stochastic eigenvalues are:
638            (  1.00000) + j(  0.0    )
639            (  0.78779) + j(  0.11539)
640            (  0.78779) + j( -0.11539)
641            (  1.00000) + j(  0.0    )
642
643      The state covariance =
644           0.000422      0.000046      0.000660     -0.002423     -0.005169
645           0.000046      0.005444      0.002287     -0.000438     -0.018554
646           0.000660      0.002287      0.002564     -0.000031     -0.010739
647          -0.002423     -0.000438     -0.000031      0.099625      0.057783
648          -0.005169     -0.018554     -0.010739      0.057783      0.166020
649
650      The expected cost =  6.61588E-04
```

```
651
652
653     For cycle  2
654
655     The feedback gain =
656        -15.139446    -5.764192   -32.346949    -7.922675     0.0
657
658     The deterministic eigenvalues are:
659        (  0.82771) + j(  0.13041)
660        (  0.82771) + j( -0.13041)
661        (  0.72586) + j(  0.17001)
662        (  0.72586) + j( -0.17001)
663
664     The stochastic eigenvalues are:
665        (  1.00000) + j(  0.0    )
666        (  0.78779) + j(  0.11539)
667        (  0.78779) + j( -0.11539)
668        (  1.00000) + j(  0.0    )
669
670     The state covariance =
671          0.000425     0.000020     0.000653    -0.002390    -0.005866
672          0.000020     0.004920     0.002266     0.000198    -0.009337
673          0.000653     0.002266     0.002575    -0.000023    -0.008107
674         -0.002390     0.000198    -0.000023     0.098878     0.047708
675         -0.005866    -0.009337    -0.008107     0.047708     0.166020
676
677     The expected cost =  6.61974E-04
678
679
680     For cycle  3
681
682     The feedback gain =
683        -15.139446    -5.764192   -32.346949    -7.922675     0.0
684
685     The deterministic eigenvalues are:
686        (  0.65331) + j(  0.21325)
687        (  0.65331) + j( -0.21325)
688        (  0.82221) + j(  0.12661)
689        (  0.82221) + j( -0.12661)
690
691     The stochastic eigenvalues are:
692        (  1.00000) + j(  0.0    )
693        (  0.78779) + j(  0.11539)
694        (  0.78779) + j( -0.11539)
695        (  1.00000) + j(  0.0    )
696
697     The state covariance =
698          0.000424    -0.000032     0.000652    -0.002331    -0.006099
699         -0.000032     0.005433     0.002377    -0.000371     0.000003
700          0.000652     0.002377     0.002578    -0.000151    -0.006013
```

```
      -0.002331    -0.000371    -0.000151     0.099538     0.036277
      -0.006099     0.000003    -0.006013     0.036277     0.166020

The expected cost =  6.61118E-04



Over one complete control period ....

The closed loop eigenvalues are:
   (  0.39914) + j(  0.29193)
   (  0.39914) + j( -0.29193)
   (  0.18972) + j(  0.26039)
   (  0.18972) + j( -0.26039)

Total cost per control period =  2.64471E-03

The average cost per cycle =  6.61176E-04



THE "SUBOPTIMAL PLANT-STATE-ONLY" REGULATOR

For cycle  0

The feedback gain =
   -11.871394    -5.477785   -33.615552    -7.738590
   -13.080427    -5.005774     2.939510     0.264790

The deterministic eigenvalues are:
   (  0.86267) + j(  0.11581)
   (  0.86267) + j( -0.11581)
   (  0.77235) + j(  0.14881)
   (  0.77235) + j( -0.14881)

The stochastic eigenvalues are:
   (  0.86267) + j(  0.11581)
   (  0.86267) + j( -0.11581)
   (  0.77235) + j(  0.14881)
   (  0.77235) + j( -0.14881)

The state covariance =
     0.000443    -0.000033     0.000695    -0.002453
    -0.000033     0.007350     0.002650    -0.002148
     0.000695     0.002650     0.002717    -0.000151
    -0.002453    -0.002148    -0.000151     0.104943
```

```
751
752     The expected cost =  6.88973E-04
753
754
755     For cycle  1
756
757     The feedback gain =
758        -13.161774    -5.647825   -32.585050    -7.616467     0.0
759
760     The deterministic eigenvalues are:
761        (  0.84567) + j(  0.12371)
762        (  0.84567) + j( -0.12371)
763        (  0.76215) + j(  0.15989)
764        (  0.76215) + j( -0.15989)
765
766     The stochastic eigenvalues are:
767        (  1.00000) + j(  0.0    )
768        (  0.79567) + j(  0.13323)
769        (  0.79567) + j( -0.13323)
770        (  1.00000) + j(  0.0    )
771
772     The state covariance =
773          0.000445     0.000048     0.000696    -0.002543    -0.005457
774          0.000048     0.005744     0.002414    -0.000481    -0.019504
775          0.000696     0.002414     0.002720    -0.000093    -0.011111
776         -0.002543    -0.000481    -0.000093     0.101446     0.061867
777         -0.005457    -0.019504    -0.011111     0.061867     0.177589
778
779     The expected cost =  6.63076E-04
780
781
782     For cycle  2
783
784     The feedback gain =
785        -14.081714    -5.720591   -31.649271    -7.524385     0.0
786
787     The deterministic eigenvalues are:
788        (  0.73093) + j(  0.18537)
789        (  0.73093) + j( -0.18537)
790        (  0.83354) + j(  0.12551)
791        (  0.83354) + j( -0.12551)
792
793     The stochastic eigenvalues are:
794        (  1.00000) + j(  0.0    )
795        (  0.79909) + j(  0.12779)
796        (  0.79909) + j( -0.12779)
797        (  1.00000) + j(  0.0    )
798
799     The state covariance =
800          0.000449     0.000021     0.000689    -0.002518    -0.006185
```

```
          0.000021     0.005190     0.002390     0.000133    -0.009627
          0.000689     0.002390     0.002721    -0.000126    -0.008298
         -0.002518     0.000133    -0.000126     0.101324     0.050899
         -0.006185    -0.009627    -0.008298     0.050899     0.177589

The expected cost =  6.47873E-04


For cycle  3

The feedback gain =
   -14.144311    -5.674044   -31.388743    -7.520733     0.0

The deterministic eigenvalues are:
   (  0.65946) + j(  0.22713)
   (  0.65946) + j( -0.22713)
   (  0.82936) + j(  0.12429)
   (  0.82936) + j( -0.12429)

The stochastic eigenvalues are:
   (  1.00000) + j(  0.0    )
   (  0.79926) + j(  0.12515)
   (  0.79926) + j( -0.12515)
   (  1.00000) + j(  0.0    )

The state covariance =
     0.000448    -0.000034     0.000688    -0.002458    -0.006416
    -0.000034     0.005716     0.002496    -0.000389     0.000381
     0.000688     0.002496     0.002720    -0.000133    -0.006068
    -0.002458    -0.000389    -0.000133     0.102818     0.038487
    -0.006416     0.000381    -0.006068     0.038487     0.177589

The expected cost =  6.48709E-04



Over one complete control period ....

The closed loop eigenvalues are:
   (  0.42152) + j(  0.28693)
   (  0.42152) + j( -0.28693)
   (  0.17306) + j(  0.28361)
   (  0.17306) + j( -0.28361)

Total cost per control period =  2.64863E-03

The average cost per cycle =  6.62158E-04
```

```
851
852
853
854      THE "AVERAGED GAINS" REGULATOR
855
856
857      For cycle  0
858
859      The feedback gain =
860          -7.723973     -4.057164    -34.855046     -7.872781
861         -13.080427     -5.005774      2.939510      0.264790
862
863      The deterministic eigenvalues are:
864         (  0.87193) + j(  0.11509)
865         (  0.87193) + j( -0.11509)
866         (  0.75470) + j(  0.14586)
867         (  0.75470) + j( -0.14586)
868
869      The stochastic eigenvalues are:
870         (  0.87193) + j(  0.11509)
871         (  0.87193) + j( -0.11509)
872         (  0.75470) + j(  0.14586)
873         (  0.75470) + j( -0.14586)
874
875      The state covariance =
876           0.000426     -0.000028      0.000676     -0.002365
877          -0.000028      0.007050      0.002512     -0.001857
878           0.000676      0.002512      0.002574     -0.000298
879          -0.002365     -0.001857     -0.000298      0.101033
880
881      The expected cost =  6.59535E-04
882
883
884      For cycle  1
885
886      The feedback gain =
887          -7.723973     -4.057164    -34.855046     -7.872781      0.255427
888
889      The deterministic eigenvalues are:
890         (  0.84600) + j(  0.12156)
891         (  0.84600) + j( -0.12156)
892         (  0.75425) + j(  0.16254)
893         (  0.75425) + j( -0.16254)
894
895      The stochastic eigenvalues are:
896         (  1.00000) + j(  0.0    )
897         (  0.78269) + j(  0.14105)
898         (  0.78269) + j( -0.14105)
899         (  1.00000) + j(  0.0    )
900
```

```
 The state covariance =
      0.000428     0.000042     0.000676    -0.002367    -0.005244
      0.000042     0.005460     0.002272     0.000002    -0.018618
      0.000676     0.002272     0.002576     0.000042    -0.010872
     -0.002367     0.000002     0.000042     0.097986     0.056134
     -0.005244    -0.018618    -0.010872     0.056134     0.170292

 The expected cost =  6.63584E-04


 For cycle  2

 The feedback gain =
     -7.723973    -4.057164   -34.855046    -7.872781     0.255427

 The deterministic eigenvalues are:
    (  0.72311) + j(  0.18714)
    (  0.72311) + j( -0.18714)
    (  0.83236) + j(  0.12415)
    (  0.83236) + j( -0.12415)

 The stochastic eigenvalues are:
    (  1.00000) + j(  0.0    )
    (  0.78269) + j(  0.14105)
    (  0.78269) + j( -0.14105)
    (  1.00000) + j(  0.0    )

 The state covariance =
      0.000431     0.000018     0.000671    -0.002363    -0.005941
      0.000018     0.004969     0.002261     0.000345    -0.009267
      0.000671     0.002261     0.002588    -0.000056    -0.008301
     -0.002363     0.000345    -0.000056     0.098302     0.046932
     -0.005941    -0.009267    -0.008301     0.046932     0.170292

 The expected cost =  6.62027E-04


 For cycle  3

 The feedback gain =
     -7.723973    -4.057164   -34.855046    -7.872781     0.255427

 The deterministic eigenvalues are:
    (  0.65772) + j(  0.22693)
    (  0.65772) + j( -0.22693)
    (  0.82462) + j(  0.12398)
    (  0.82462) + j( -0.12398)

 The stochastic eigenvalues are:
    (  1.00000) + j(  0.0    )
```

```
      (  0.78269) + j(  0.14105)
      (  0.78269) + j( -0.14105)
      (  1.00000) + j(  0.0    )

The state covariance =
     0.000430    -0.000031     0.000670    -0.002338    -0.006167
    -0.000031     0.005499     0.002367    -0.000277     0.000219
     0.000670     0.002367     0.002587    -0.000190    -0.006226
    -0.002338    -0.000277    -0.000190     0.099333     0.036250
    -0.006167     0.000219    -0.006226     0.036250     0.170292

The expected cost =  6.60331E-04



Over one complete control period ....

The closed loop eigenvalues are:
   (  0.42501) + j(  0.29313)
   (  0.42501) + j( -0.29313)
   (  0.16616) + j(  0.26752)
   (  0.16616) + j( -0.26752)

Total cost per control period =  2.64548E-03

The average cost per cycle =  6.61369E-04




THE "SEPARATED MODES" MULTIRATE REGULATOR


For cycle  0

The feedback gain =
   -13.225744    -5.678149   -32.648042    -7.613162
   -12.943383    -4.957848     3.453492     0.354652

The deterministic eigenvalues are:
   (  0.86113) + j(  0.11762)
   (  0.86113) + j( -0.11762)
   (  0.77719) + j(  0.14623)
   (  0.77719) + j( -0.14623)

The stochastic eigenvalues are:
   (  0.86113) + j(  0.11762)
   (  0.86113) + j( -0.11762)
```

```
      (  0.77719) + j(  0.14623)
      (  0.77719) + j( -0.14623)

 The state covariance =
      0.000522    -0.000034     0.000708    -0.002647
     -0.000034     0.008977     0.002868    -0.003902
      0.000708     0.002868     0.002593    -0.000324
     -0.002647    -0.003902    -0.000324     0.105590

 The expected cost =  6.61299E-04


 For cycle  1

 The feedback gain =
   -13.225744    -5.678149   -32.648042    -7.613162     0.0

 The deterministic eigenvalues are:
   (  0.84678) + j(  0.12287)
   (  0.84678) + j( -0.12287)
   (  0.75975) + j(  0.16233)
   (  0.75975) + j( -0.16233)

 The stochastic eigenvalues are:
   (  1.00000) + j(  0.0    )
   (  0.79579) + j(  0.13401)
   (  0.79579) + j( -0.13401)
   (  1.00000) + j(  0.0    )

 The state covariance =
      0.000525     0.000061     0.000707    -0.002753    -0.006543
      0.000061     0.005847     0.002566    -0.001270    -0.022928
      0.000707     0.002566     0.002592     0.000005    -0.010444
     -0.002753    -0.001270     0.000005     0.102370     0.074180
     -0.006543    -0.022928    -0.010444     0.074180     0.223688

 The expected cost =  6.62731E-04


 For cycle  2

 The feedback gain =
   -13.225744    -5.678149   -32.648042    -7.613162     0.0

 The deterministic eigenvalues are:
   (  0.72230) + j(  0.19136)
   (  0.72230) + j( -0.19136)
   (  0.83661) + j(  0.12379)
   (  0.83661) + j( -0.12379)
```

```
1051      The stochastic eigenvalues are:
1052         (  1.00000) + j(  0.0    )
1053         (  0.79579) + j(  0.13401)
1054         (  0.79579) + j( -0.13401)
1055         (  1.00000) + j(  0.0    )
1056
1057      The state covariance =
1058           0.000529     0.000021     0.000699    -0.002698    -0.007369
1059           0.000021     0.006139     0.002558    -0.000406    -0.010162
1060           0.000699     0.002558     0.002605    -0.000009    -0.007178
1061          -0.002698    -0.000406    -0.000009     0.101373     0.056780
1062          -0.007369    -0.010162    -0.007178     0.056780     0.223688
1063
1064      The expected cost =  6.63083E-04
1065
1066
1067      For cycle  3
1068
1069      The feedback gain =
1070         -13.225744    -5.678149   -32.648042    -7.613162     0.0
1071
1072      The deterministic eigenvalues are:
1073         (  0.64886) + j(  0.23588)
1074         (  0.64886) + j( -0.23588)
1075         (  0.83155) + j(  0.12320)
1076         (  0.83155) + j( -0.12320)
1077
1078      The stochastic eigenvalues are:
1079         (  1.00000) + j(  0.0    )
1080         (  0.79579) + j(  0.13401)
1081         (  0.79579) + j( -0.13401)
1082         (  1.00000) + j(  0.0    )
1083
1084      The state covariance =
1085           0.000527    -0.000046     0.000698    -0.002621    -0.007557
1086          -0.000046     0.006857     0.002704    -0.001330     0.002644
1087           0.000698     0.002704     0.002607    -0.000174    -0.004790
1088          -0.002621    -0.001330    -0.000174     0.102560     0.038940
1089          -0.007557     0.002644    -0.004790     0.038940     0.223688
1090
1091      The expected cost =  6.62152E-04
1092
1093
1094
1095
1096      Over one complete control period ....
1097
1098      The closed loop eigenvalues are:
1099         (  0.42654) + j(  0.28601)
1100         (  0.42654) + j( -0.28601)
```

```
1101          (  0.15871) + j(  0.28510)
1102          (  0.15871) + j( -0.28510)
1103
1104       Total cost per control period =  2.64926E-03
1105
1106       The average cost per cycle =  6.62316E-04
End of file
```